A Model-Free Control System Based on the Sliding Mode Control Method with Applications to Multi-Input-Multi-Output Systems

Fares El Tin
fse5947@rit.edu

Follow this and additional works at: http://scholarworks.rit.edu/theses

Recommended Citation

This Thesis is brought to you for free and open access by the Thesis/Dissertation Collections at RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.
A Model-Free Control System Based on the Sliding Mode Control Method with Applications to Multi-Input-Multi-Output Systems

By

Fares El Tin

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Mechanical Engineering

Supervised by

Dr. Agamemnon Crassidis

Department of Mechanical Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York
May 2017

Approved by

Dr. Agamemnon Crassidis, Professor
Thesis Advisor, Department of Mechanical Engineering

Dr. Mark Kempski, Professor
Committee Member, Department of Mechanical Engineering

Dr. Jason Kolodziej, Associate Professor
Committee Member, Department of Mechanical Engineering

Dr. Wayne Walter, Professor
Department Representative, Mechanical Engineering
Acknowledgements

I would like to express my sincere gratitude to my advisor Dr. Agamemnon Crassidis for his continued support and motivation. I would also like to thank my thesis committee: Dr. Mark Kempski, Dr. Jason Kolodziej, and Dr. Wayne Walter; their immense knowledge and motivation stirred my engineering curiosity and inspired me to pursue research in the field of controls.

Most importantly, this would not have been possible without the endless support and patience of my family.

Finally, I would like to thank the staff at the department of Mechanical Engineering and my friends here at RIT who made me feel at home away from home.
Abstract

In this work, a model-free sliding mode control technique for linear and nonlinear uncertain multi-input multi-output systems is proposed. The developed method does not require a mathematical model of the dynamic system. Instead, knowledge of the system’s order, state measurements, and control input gain matrix shape and bounds are assumed to develop the control law and drive the system’s states to track a desired trajectory. The control system relies on estimating the error between previous and current control inputs to stabilize the system. Lyapunov’s stability criterion is used in the derivation process to ensure closed-loop asymptotic stability. High frequency chattering of the control input and higher-order states, often observed with the sliding mode control method, is eliminated using a smoothing boundary layer. Simulations are performed on a variety of linear and nonlinear systems, including a quadrotor model, to test the performance of the control law. Finally, the model-free sliding mode control system is modified to account for the effects of actuator time-delays.
# Table of Contents

ACKNOWLEDGEMENTS .................................................................................................................. II

ABSTRACT ................................................................................................................................... III

LIST OF FIGURES ......................................................................................................................... VII

NOMENCLATURE ............................................................................................................................. XIV

1 INTRODUCTION .......................................................................................................................... 1

1.1 BACKGROUND .......................................................................................................................... 1

1.2 LITERATURE REVIEW ............................................................................................................. 2

1.2.1 Classical SMC Schemes ...................................................................................................... 2

1.2.2 Model-free SMC Schemes ................................................................................................. 5

1.3 GAPS IN PREVIOUS RESEARCH ON MODEL-FREE SMC .................................................... 8

1.4 RESEARCH GOALS .................................................................................................................. 8

2 FUNDAMENTALS OF THE LYAPUNOV THEORY .................................................................. 10

2.1 NONLINEAR SYSTEMS AND EQUILIBRIUM POINTS ............................................................ 11

2.1.1 Autonomous and Non-autonomous systems ...................................................................... 11

2.1.2 Equilibrium Points ............................................................................................................. 12

2.2 CONCEPTS OF STABILITY .................................................................................................... 13

2.3 LYAPUNOV’S DIRECT METHOD ......................................................................................... 14

2.4 POSITIVE DEFINITE FUNCTIONS ......................................................................................... 16

3 SLIDING MODE CONTROL ....................................................................................................... 18

3.1 DERIVATION OF THE SLIDING MODE CONTROL METHODOLOGY FOR MIMO SYSTEMS .. 18

3.2 DEFINING THE BOUNDARY LAYER ...................................................................................... 21

3.3 ILLUSTRATIVE EXAMPLE .................................................................................................... 23

4 THE MODEL-FREE SMC SCHEME FOR MIMO SYSTEMS ...................................................... 31
4.1 The Definition of Square and Non-square MIMO Systems .......................................................... 31
4.2 Model-free SMC for Square MIMO Systems .............................................................................. 32
  4.2.1 System Description ................................................................................................................ 33
4.3 First-order Square MIMO System Control Law ...................................................................... 34
  4.3.1 Asymptotic Stability of the Controller .................................................................................. 34
  4.3.2 The Switching Gain .............................................................................................................. 35
  4.3.3 The Boundary Layer ............................................................................................................ 37
  4.3.4 Illustrative Example ............................................................................................................. 37
4.4 Second-order Square MIMO System Control Law ................................................................. 41
  4.4.1 Asymptotic Stability of the Controller .................................................................................. 41
  4.4.2 The Switching Gain .............................................................................................................. 42
  4.4.3 The Boundary Layer ............................................................................................................ 44
  4.4.4 Illustrative Example ............................................................................................................. 44
4.4 Model-free SMC for Non-square MIMO Systems .................................................................... 48
  4.4.1 System Description ................................................................................................................ 49
4.5 First-order Non-square MIMO System Control Law ............................................................... 50
  4.5.1 Asymptotic Stability of the Controller .................................................................................. 51
  4.5.2 The Switching Gain .............................................................................................................. 52
  4.5.3 The Boundary Layer ............................................................................................................ 53
  4.5.4 Illustrative Example ............................................................................................................. 54
4.6 Second-order Non-square MIMO System Control Law ........................................................... 59
  4.6.1 Asymptotic Stability of the Controller .................................................................................. 60
  4.6.2 The Switching Gain .............................................................................................................. 61
  4.6.3 The Boundary Layer ............................................................................................................ 62
  4.6.4 Illustrative Example ............................................................................................................. 63
4.7 Position Control of a Quadrotor ............................................................................................... 70
4.8 The Transformation Matrix ..................................................................................................... 74
List of Figures

Figure 2.1: A graphical representation of the classic pendulum problem. .................................................12

Figure 2.2: An illustration of the concepts of stability [1]. ........................................................................14

Figure 2.3: A nonlinear mass-spring-damper system. .............................................................................15

Figure 2.4: Lyapunov surfaces illustrating the reasoning behind the radial unboundedness condition [1]. .17

Figure 3.1: A graphical representation of the SMC process. $x_d(t)$ is the desired state of the system and $s$, which will be described later, is the sliding surface.................................................................19

Figure 3.2: A graphical representation of chattering along the sliding surface. ........................................21

Figure 3.3: The boundary layer introduced neighboring the sliding surface. $\varphi$ is the thickness of the boundary layer, and $\epsilon$ is the boundary layer width. .................................................................21

Figure 3.4: Left: A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. Right: The tracking error, $x_1-x_{1d}$, shows miniscule error, in the order of $10^{-3}$. .................................................................26

Figure 3.5: Left: A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. Right: The tracking error, $x_2-x_{2d}$, shows miniscule error, in the order of $10^{-3}$. .................................................................27

Figure 3.6: Left: A comparison of the 1st derivative of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. Right: The 1st derivative tracking error, $\dot{x}_1-\dot{x}_{1d}$, shows considerable error, in the order of $10^0$. ..........26

Figure 3.7: Left: A comparison of the 1st derivative of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. Right: The 1st derivative tracking error, $\dot{x}_2-\dot{x}_{2d}$, shows considerable error, in the order of $10^0$.............27

Figure 3.8: Left: The controller effort $u_1$. Right: The controller effort $u_2$..................................................27

Figure 3.9: Left: The sliding surface condition of controller 1. Right: The sliding surface condition of controller 2 ........................................................................................................................................27

Figure 3.10: Left: A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. Right: The tracking error, $x_1-x_{1d}$, shows miniscule error, in the order of $10^{-2}$. .................................................................28

Figure 3.11: Left: A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. Right: The tracking error, $x_2-x_{2d}$, shows miniscule error, in the order of $10^{-2}$. .................................................................28
Figure 3.12: **Left:** A comparison of the 1st derivative of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. 

**Right:** The 1st derivative tracking error, $\dot{x}_1 - \dot{x}_{1d}$, shows minimal error, in the order of $10^{-2}$. ......................................................... 29

Figure 3.13: **Left:** A comparison of the 1st derivative of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. 

**Right:** The 1st derivative tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows minimal error, in the order of $10^{-2}$. ......................................................... 29

Figure 3.14: **Left:** The controller effort $u_1$. **Right:** The controller effort $u_2$. ................................................................. 30

Figure 3.15: **Left:** The sliding surface condition of controller 1. **Right:** The sliding surface condition of controller 2. ......................................................................................................................... 30

Figure 4.1: **Left:** A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. **Right:** The tracking error, $x_1 - x_{1d}$, shows miniscule error, in the order of $10^{-5}$. ................................................................. 38

Figure 4.2: **Left:** A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. **Right:** The tracking error, $x_2 - x_{2d}$, shows miniscule error, in the order of $10^{-7}$. ................................................................. 38

Figure 4.3: **Left:** A comparison of the 1st derivative of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. 

**Right:** The 1st derivative tracking error, $\dot{x}_1 - \dot{x}_{1d}$, shows miniscule error, in the order of $10^{-5}$, following the initial spike. ......................................................................................................................... 39

Figure 4.4: **Left:** A comparison of the 1st derivative of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. 

**Right:** The 1st derivative tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows miniscule error, in the order of $10^{-5}$, following the initial spike. ......................................................................................................................... 39

Figure 4.5: **Left:** The controller effort $u_1$. **Right:** The controller effort $u_2$. ................................................................. 40

Figure 4.6: The boundary layers of both controllers’ sliding surfaces. ................................................................. 40

Figure 4.7: A fully-actuated 2 mass-spring-damper system. ................................................................. 44

Figure 4.8: **Left:** A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. **Right:** The velocity tracking error, $x_1 - x_{1d}$, shows miniscule error, in the order of $10^{-6}$. ................................................................. 46

Figure 4.9: **Left:** A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. **Right:** The velocity tracking error, $x_2 - x_{2d}$, shows miniscule error, in the order of $10^{-6}$. ................................................................. 46

VIII
Figure 4.10: **Left:** A comparison of the velocity of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$. **Right:** The velocity tracking error, $\dot{x}_1 - \dot{x}_{1d}$, shows miniscule error, in the order of $10^{-6}$. .................................................................46

Figure 4.11: **Left:** A comparison of the velocity of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$. **Right:** The velocity tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows miniscule error, in the order of $10^{-6}$. .................................................................46

Figure 4.12: **Left:** A comparison of the acceleration of output $x_1(t)$ and the desired trajectory $\ddot{x}_{1d}(t)$. **Right:** The acceleration tracking error, $\ddot{x}_1 - \ddot{x}_{1d}$, shows miniscule error, in the order of $10^{-4}$, following the initial spike.................................................................47

Figure 4.13: **Left:** A comparison of the acceleration of output $x_2(t)$ and the desired trajectory $\ddot{x}_{2d}(t)$. **Right:** The acceleration tracking error, $\ddot{x}_2 - \ddot{x}_{2d}$, shows miniscule error, in the order of $10^{-4}$, following the initial spike.................................................................47

Figure 4.14: **Left:** The controller effort $u_1$. **Right:** The controller effort $u_2$ .................................................................47

Figure 4.15: **Left:** The boundary layer of $s_1$. **Right:** The boundary layer of $s_2$ .................................................................48

Figure 4.16: **Left:** A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. **Right:** $x_1(t)$ tracking error, $x_1 - x_{1d}$, shows miniscule error, in the order of $10^2$. .................................................................55

Figure 4.17: **Left:** A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. **Right:** $x_2(t)$ tracking error, $x_2 - x_{2d}$, shows significant error, in the order of $10^0$. .................................................................55

Figure 4.18: **Left:** A comparison of the 1st derivative of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$. **Right:** The 1st derivative tracking error, $\dot{x}_1 - \dot{x}_{1d}$, shows miniscule error, in the order of $10^2$, following the initial spike.................................................................56

Figure 4.19: **Left:** A comparison of the 1st derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$. **Right:** The 1st derivative tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows significant error, in the order of $10^0$. .................................................................56

Figure 4.20: **Left:** A verification of the sliding condition. **Right:** The controller effort. .................................................................56

Figure 4.21: **Left:** A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. **Right:** $x_1(t)$ tracking error, $x_1 - x_{1d}$, shows significant error, in the order of $10^0$. .................................................................57

Figure 4.22: **Left:** A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. **Right:** $x_2(t)$ tracking error, $x_2 - x_{2d}$, shows miniscule error, in the order of $10^2$. .................................................................57
Figure 4.23: Left: A comparison of the 1\textsuperscript{st} derivative of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$. Right: The 1\textsuperscript{st} derivative tracking error, $x_1 - \dot{x}_{1d}$, shows significant error, in the order of $10^0$, following the initial spike.

Figure 4.24: Left: A comparison of the 1\textsuperscript{st} derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$. Right: The 1\textsuperscript{st} derivative tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows miniscule error, in the order of $10^{-2}$, following the initial spike.

Figure 4.25: Left: A verification of the sliding condition in Eq. (5.4). Right: The controller effort.

Figure 4.26: An underactuated 2 mass-spring-damper system.

Figure 4.27: Left: A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. Right: $x_1(t)$ tracking error, $x_1 - x_{1d}$, shows minimal error, in the order of $10^{-2}$.

Figure 4.28: Left: A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. Right: $x_2(t)$ tracking error, $x_2 - x_{2d}$, shows considerable error, in the order of $10^{-1}$.

Figure 4.29: Left: A comparison of the velocity of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$. Right: The velocity tracking error, $\dot{x}_1 - \dot{x}_{1d}$, shows miniscule error, in the order of $10^{-2}$.

Figure 4.30: Left: A comparison of the velocity of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$. Right: The velocity tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows some error, in the order of $10^{-1}$.

Figure 4.31: Left: A comparison of the acceleration of output $x_1(t)$ and the desired trajectory $\ddot{x}_{1d}(t)$. Right: The acceleration tracking error, $\ddot{x}_1 - \ddot{x}_{1d}$, shows miniscule error, in the order of $10^{-2}$, following the initial spike.

Figure 4.32: Left: A comparison of the acceleration of output $x_2(t)$ and the desired trajectory $\ddot{x}_{2d}(t)$. Right: The acceleration tracking error, $\ddot{x}_2 - \ddot{x}_{2d}$, shows significant error, in the order of $10^{1}$, following the initial spike.

Figure 4.33: Left: A verification of the sliding condition. Right: The controller effort.

Figure 4.34: Left: A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. Right: $x_1(t)$ tracking error, $x_1 - x_{1d}$, shows considerable error, in the order of $10^{-1}$. 
Figure 4.35: **Left**: A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. **Right**: $x_2(t)$ tracking error, $x_2-x_{2d}$, shows minimal error, in the order of $10^{-2}$.

Figure 4.36: **Left**: A comparison of the velocity of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$. **Right**: The velocity tracking error, $\dot{x}_1-\dot{x}_{1d}$, shows some error, in the order of $10^{-1}$.

Figure 4.37: **Left**: A comparison of the velocity of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$. **Right**: The velocity tracking error, $\dot{x}_2-\dot{x}_{2d}$, shows miniscule error, in the order of $10^{-2}$.

Figure 4.38: **Left**: A comparison of the acceleration of output $x_1(t)$ and the desired trajectory $\ddot{x}_{1d}(t)$. **Right**: The acceleration tracking error, $\ddot{x}_1-\ddot{x}_{1d}$, shows significant error, in the order of $10^{-1}$.

Figure 4.39: **Left**: A comparison of the acceleration of output $x_2(t)$ and the desired trajectory $\ddot{x}_{2d}(t)$. **Right**: The acceleration tracking error, $\ddot{x}_2-\ddot{x}_{2d}$, shows miniscule error, in the order of $10^{-2}$.

Figure 4.40: **Left**: A verification of the sliding condition. **Right**: The controller effort.

Figure 4.41: **Left**: Plots of the position of the quadrotor compared to the desired trajectory.

Figure 4.42: **Left**: Plot of the altitude of the quadrotor. **Right**: Plot of the heading of the quadrotor.

Figure 4.43: **Left**: Plot of the pitch angle of the quadrotor. **Right**: Plot of the roll angle of the quadrotor.

Figure 4.44: Plots of the control inputs $u_1$ and $u_2$.

Figure 4.45: Plots of the control inputs $u_3$ and $u_4$.

Figure 5.1: **Left**: A comparison of output $x(t)$ and the desired trajectory $x_d(t)$. **Right**: $x(t)$ tracking error, $x-x_d$, shows miniscule error, in the order of $10^{-4}$.

Figure 5.2: **Left**: A comparison of the 1st derivative of output $x(t)$ and the desired trajectory $\dot{x}_d(t)$. **Right**: The 1st derivative tracking error, $\dot{x}-\dot{x}_d$, begins in the order of $10^{-1}$, but then quickly drops to the order of $10^{-3}$.

Figure 5.3: **Left**: A verification of the boundary layer condition. **Right**: The controller effort.

Figure 5.4: **Left**: A comparison of output $x(t)$ and the desired trajectory $x_d(t)$. **Right**: $x(t)$ tracking error, $x-x_d$, shows miniscule error, in the order of $10^{-2}$.

Figure 5.5: **Left**: A comparison of the 1st derivative of output $x(t)$ and the desired trajectory $\dot{x}_d(t)$. **Right**: The 1st derivative tracking error, $\dot{x}-\dot{x}_d$, in the order of $10^{-2}$.

Figure 5.6: **Left**: The boundary layer condition. **Right**: The controller effort.
Figure 5.7: A verification of the boundary layer condition. .................................................................82

Figure 5.8: **Left**: A comparison of output \( x(t) \) and the desired trajectory \( x_d(t) \). **Right**: \( x(t) \) tracking error, \( x-x_d \), shows miniscule error, in the order of \( 10^{-3} \). .................................................................83

Figure 5.9: **Left**: A comparison of the 1\textsuperscript{st} derivative of output \( x(t) \) and the desired trajectory \( \dot{x}_d(t) \). **Right**: The 1\textsuperscript{st} derivative tracking error, \( \dot{x}-\dot{x}_d \), in the order of \( 10^{-2} \) ........................................................................84

Figure 5.10: **Left**: A comparison of the 2\textsuperscript{nd} derivative of output \( x(t) \) and the desired trajectory \( \ddot{x}_d(t) \). **Right**: The 2\textsuperscript{nd} derivative tracking error, \( \ddot{x}-\ddot{x}_d \) in the order of \( 10^{-1} \) ........................................................................84

Figure 5.11: **Left**: A verification of the boundary layer condition. **Right**: The controller effort. .................84

Figure 5.12: **Left**: A comparison of output \( x_1(t) \) and the desired trajectory \( x_{1d}(t) \). **Right**: \( x_1(t) \) tracking error, \( x_1-x_{1d} \), shows miniscule error, in the order of \( 10^{-3} \), following the initial spike. .........................86

Figure 5.13: **Left**: A comparison of output \( x_2(t) \) and the desired trajectory \( x_{2d}(t) \). **Right**: \( x_2(t) \) tracking error, \( x_2-x_{2d} \), shows miniscule error, in the order of \( 10^{-3} \), following the initial spike.........................86

Figure 5.14: **Left**: A comparison of the 1\textsuperscript{st} derivative of output \( x_1(t) \) and the desired trajectory \( \dot{x}_{1d}(t) \).  

**Right**: The 1\textsuperscript{st} derivative tracking error, \( \dot{x}_1-\dot{x}_{1d} \), shows miniscule error, in the order of \( 10^{-3} \), following the initial spike.........................................................................................87

Figure 5.15: **Left**: A comparison of the 1\textsuperscript{st} derivative of output \( x_2(t) \) and the desired trajectory \( \dot{x}_{2d}(t) \).  

**Right**: The 1\textsuperscript{st} derivative tracking error, \( \dot{x}_2-\dot{x}_{2d} \), shows miniscule error, in the order of \( 10^{-3} \), following the initial spike...............................................................87

Figure 5.16: A verification of the sliding condition of the **Left**: 1\textsuperscript{st} output, **Right**: 2\textsuperscript{nd} output. ...............87

Figure 5.17: **Left**: The controller effort \( u_1 \). **Right**: The controller effort \( u_2 \). .................................88

Figure 5.18: **Left**: A comparison of output \( x_1(t) \) and the desired trajectory \( x_{1d}(t) \). **Right**: \( x_1(t) \) tracking error, \( x_1-x_{1d} \), shows miniscule error, in the order of \( 10^{-2} \). .................................................................89

Figure 5.19: **Left**: A comparison of output \( x_2(t) \) and the desired trajectory \( x_{2d}(t) \). **Right**: \( x_2(t) \) tracking error, \( x_2-x_{2d} \), shows significant error, in the order of \( 10^0 \). ........................................................................90

XII
Figure 5.20: **Left**: A comparison of the 1st derivative of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$.

**Right**: The 1st derivative tracking error, $\dot{x}_1 - \dot{x}_{1d}$, shows miniscule error, in the order of $10^{-2}$, following the initial spike.

Figure 5.21: **Left**: A comparison of the 1st derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$.

**Right**: The 1st derivative tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows significant error, in the order of $10^0$, following the initial spike.

Figure 5.22: **Left**: A verification of the boundary layer condition. **Right**: The controller effort.

Figure 5.23: **Left**: A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. **Right**: $x_1(t)$ tracking error, $x_1 - x_{1d}$, shows significant error, in the order of $10^0$, following the initial spike.

Figure 5.24: **Left**: A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. **Right**: $x_2(t)$ tracking error, $x_2 - x_{2d}$, shows miniscule error, in the order of $10^{-2}$, following the initial spike.

Figure 5.25: **Left**: A comparison of the 1st derivative of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$.

**Right**: The 1st derivative tracking error, $\dot{x}_1 - \dot{x}_{1d}$, shows significant error, in the order of $10^0$, following the initial spike.

Figure 5.26: **Left**: A comparison of the 1st derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$.

**Right**: The 1st derivative tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows miniscule error, in the order of $10^{-2}$, following the initial spike.

Figure 5.27: **Left**: A verification of the boundary layer condition. **Right**: The controller effort.
Nomenclature

UAV – Unmanned Aerial Vehicle
PID – Proportional Integral Derivative
SMC – Sliding Mode Control
φ – Roll
θ – Pitch
ψ – Yaw
TORA – Translational Oscillator with Rotational Actuator
SMO – Sliding Mode Observer
Sliding-PD – Sliding Proportional Derivative
DOF – Degree of Freedom
VF – Velocity Field
SISO – Single-Input Single-Output
MIMO – Multi-Input Multi-Output
\( \vec{x}_p \) – System Output Vector
n – Order of the System
\( u_m \) – Control Input
\( f_p(x; u_m; t) \) – Continuous Function of \( x \)
t – Time
p – # of outputs
m – # of inputs
\( \vec{x} \) – Vector of the System’s States
\([A]\) – \( nxn \) State Matrix
\([B]\) – \( n xm \) Matrix
\( M \) – Mass of the Pendulum
\( R \) – Length of the Pendulum – Radius of Sphere
\( \theta \) – Angle between the Pendulum and the Vertical
\( b \) – Friction at the Pivot Point
\( g \) – Acceleration of Gravity
\( r \) – radius
\( x_e \) – Equilibrium Point
\( \alpha \) – Strictly Positive Number
\( \delta \) – Strictly Positive number
m – mass
\( x \) – System State
\( c \) – Damping Coefficient
\( k \) – Spring Constant
\( b \) – Spring Stiffening Coefficient
\( V(x) \) – Lyapunov Function
\([B]_{p \times m} \) – Control Input Gain
\( x_{d_p} \) – Desired State
\( s_m \) – Sliding Surface
\( \lambda_m \) – Vector of Positive Constants
\( \tilde{x}_p \) – Tracking Error
\( \eta_m \) – Vector of Small Positive Constants
\( \hat{u}_m \) – Control Input Estimation
\( K_m \) – Switching Gain
\( sgn() \) – Signum Function
\( \varphi_m \) – Boundary Layer Thickness
\( \varepsilon \) – Boundary Layer Width
\( \bar{K}_m \) – New Switching Gain
\( \text{sat}(\cdot) \) – Saturation Function
\( \alpha_1 \) – Uncertain Time-varying Parameter
\( \alpha_2 \) – Uncertain Time-varying Parameter
\( x_{d_1}(t) \) – Desired Trajectory of First State
\( x_{d_2}(t) \) – Desired Trajectory of Second State
\( \hat{\alpha}_1 \) – Estimate of Parameter
\( \hat{\alpha}_2 \) – Estimate of Parameter
\( u_{m_{k-1}} \) – Previous Control Input
\( b_{\text{lower}} \) – Lower Bound of Control Input Gain Element
\( b_{\text{mm}} \) – Control Input Gain Element
\( b_{\text{upper}} \) – Upper Bound of Control Input Gain Element
\( \varepsilon_m \) – Control Input Error
\( \hat{\varepsilon}_m \) – Control Input Error Estimation
\( u_{m_{k-2}} \) – Previous Control Input of the Previous Control Input
\( \sigma_l \) – Lower Bound of Estimation Error
\( \sigma_u \) – Upper Bound of Estimation Error
\( \dot{x} \) – First Derivative of Tracking Error
\( \dot{s} \) – First Derivative of Sliding Surface
\( \ddot{x} \) – Second Derivative of Tracking Error
\([\beta] \) – Control Input Gain Auxiliary Variable Matrix
\( m_i \) – Mass
\( x_i \) – Output
\( u_i \) – Input
\( c_i \) – Damping Coefficient
\( k_i \) – Spring Constant
\( \beta_i \) – Spring Stiffening Coefficient
$i$ – Number of Masses
$y$ – Transformed Coordinate
$[T]$ – Transformation Matrix
$T_{11}$ – Transformation Matrix Element
$T_{12}$ – Transformation Matrix Element
$(x, y, z)$ – Position Coordinates
$(\theta, \psi, \varphi)$ – Euler Angles Representing Pitch, Roll, and Yaw
$K_t$ – Drag Coefficients
$l$ – Half the Length of the Quadroto
$I_t$ – Moments of Inertia
$u_t$ – Virtual Control Inputs
$F_t$ – Rotor Thrust
$C$ – Force to Moment Scaling Factor
$(x_d, y_d, z_d, \varphi_d)$ – Desired Conditions
$J$ – Cost Function
$t_f$ – Simulation Run Time
$k_i$ – Weighing Scale
$H(s)$ – Actuator Time Delay Transfer Function
$\tau$ – Actuator Time Constant
$u_{pa}$ – Control Input Post Actuator
$b$ – Control Input Post Actuator Gain
$a$ – Control Input Post Actuator Gain
1 Introduction

1.1 Background

Advancements in the field of control systems are on the rise. From autonomous ground and aerial vehicles, to Unmanned Aerial Vehicles (UAVs), system automation is becoming more stable and versatile every day. The most common of control theories is the Proportional-Integral-Derivative (PID) control system, which drives systems to a desired state by compensating for errors. However, due to its theoretical limitations to strictly certain linear or linearizable systems, in addition to requiring a precise system model to be applied to most physical systems, performance, in some instances, suffers.

In nonlinear control theory, Sliding Mode Control (SMC) is a robust control methodology for both linear and nonlinear systems with modeling uncertainties. The premise of the technique is that it is much easier to control 1st-order systems of any type, than it is to control higher order systems [1]. Therefore, by transforming the control problem into a 1st-order problem, "perfect" performance is easier to achieve. The method breaks the control problem down into two phases, a reaching phase and a sliding phase. The reaching phase drives the system states towards the sliding surface, where the sliding phase reacts and slides the states towards equilibrium. By remaining on the sliding surface, asymptotic stability, in the Lyapunov sense, is guaranteed. SMC promises to be an extremely powerful control tool, however, the classic SMC methodology requires a mathematical model of the system in question, and is unique to each system, which restricts its use on that basis.
1.2 Literature Review

SMC has been well-researched and applied to various systems, such as robotic manipulators, power systems, and unmanned vehicles. In this literature review, previous work performed on the SMC system is presented. The chapter is organized as follows: the first section focuses on the conventional use of SMC, where a system model is required. Research on Discrete-time SMC applications are also presented. The second section examines recent work performed involving the model-free SMC approach in controlling linear and nonlinear systems. The final section considers the gaps in previous research in comparison to the proposed work.

1.2.1 Classical SMC Schemes

UAVs, have received a big boost in popularity in recent years for their wide range of possible applications and ease of use. Most UAVs rely on a PID control law, which provides a good stable response despite the presence of external and internal errors. However, UAV dynamics are non-linear and require linearization to compute optimum PID control gains, which can affect performance [2]. In addition, the controller gains are specific to the platform's size and weight, and need to be tuned for each vehicle individually and following any change to the overall system characteristics. Therefore, researchers have studied the use of nonlinear control methods, such as feedback linearization and SMC, on UAVs.

Runcharoong and Srirachatrapimuk [3] designed a SMC system for attitude control of a quadrotor. Attitude is defined as $\phi =$ roll, $\theta =$ pitch, $\psi =$ yaw, which are referred to as the Euler angles and describe the orientation of the quadrotor. A PD controller was used for altitude, $z$, and position, $x$ and $y$, control. The equations describing the dynamics of position and altitude were
linearized by the authors in order to quantify the PD control gains. Assumptions such as $\varphi \approx \psi \approx 0$ in the x-axis, $\theta \approx \psi \approx 0$ in the y-axis, and $\varphi \approx \psi \approx 0$ in the z-axis, were used to do so. Basic SMC method, with the addition of a boundary layer, to eliminate chattering about the sliding surface, was applied to the dynamical equations of the Euler angles. A simulation proved the stability of the system and the developed control inputs were able to drive the quadrotor to the desired position with the desired orientation.

The authors in [3] applied a combination of SMC and PD control on the derived system to achieve a stable output. The method, although an improvement to the typical PID control used on quadrotors, still limits the full potential of the SMC method, which does not require any linearization. The reason the authors used two different control methods on the dynamic system of the quadrotor is the presence of underactuated states, where the number of inputs is less than the number of outputs, which require further manipulation to be used in the SMC process.

Xu and Ozguner [4] proposed an approach to stabilizing underactuated systems using SMC. The system is transformed into a cascade normal form, utilizing a systematic method proposed by Olfati-Saber [5], before being used in the design of the controller. The authors applied this approach to two examples of nonlinear underactuated MIMO systems, a translational oscillator with rotational actuator (TORA), and a quadrotor UAV. The quadrotor system model used was similar to the one used by Runcharoon and Srichatrapimuk in [3]. To begin the controller design, the system model was divided into a fully actuated subsystem, consisting of the equations describing $z$ and $\psi$, and an underactuated subsystem, composed of the remaining parameters. To control the former, the authors constructed a rate bounded PID controller and a sliding mode controller [6]. As for the latter, a transformation was applied to the system and the
SMC method was used to stabilize the subsystem. Finally, a simulation of the control law resulted in a stable system and the quadrotor converged to its desired position.

Sen et al. [7] extended the work in [4] by proposing an adaptive technique based on SMC used in quadrotor stabilization. SMC can be extremely powerful when uncertainties exist in a system, requiring only the knowledge of the bounds on these uncertainties, to be designed. However, since it is often difficult to estimate such bounds accurately, controller effort is maximized as a result of overestimation by the designer. Therefore, in order to reduce controller activity, the authors introduced and proved the stability of an adaptive law that defines a controller gain coefficient. A simulation of the designed adaptive SMC law was applied to the quadrotor system model and good stable tracking was achieved with minimized controller effort and no prior knowledge of uncertainty bounds.

In addition to controlling continuous-time systems, SMC has also demonstrated robust tracking in the realm of discrete-time systems. Pai [8] applied a discrete-time control scheme, based on discrete-time integral SMC, on uncertain linear systems, to track a desired reference signal. The author introduced an auxiliary control function to design the discrete-time sliding mode controller and stabilize the system. The switching surface of the control law was designed by extending the concept of integral switching function from continuous-time SMC to discrete-time SMC, and then completed the control law design such that quasi-sliding mode is reached. The author did note that in practice, discrete-time SMC systems can only approach the switching surface, and not stay on it, therefore only the quasi-sliding mode is assured [9,10]. The author applied the designed controller to a discrete-time system, and stability of the closed-loop system was proven while achieving outstanding tracking performance in the presence of uncertainties. Due to the integral switching surface, which the author used in the design process, the reaching
phase was also successfully eliminated. Finally, since the discrete-time SMC law introduced did not require a switching sign, no chattering was observed.

As shown in the above review, SMC has been a focus of control systems research, and its performance has been proven to produce outstanding tracking and stability of uncertain continuous-time and discrete-time, linear and nonlinear systems. The following section reviews research done on model-free SMC methods.

1.2.2 Model-free SMC Schemes

As mentioned earlier, a model-free approach to designing a sliding mode controller can be extremely beneficial, especially when dealing with complex dynamical systems.

Martinez-Guerra et al. [11] proposed a Sliding Mode Observer (SMO) to solve a certain type of synchronization problem of chaotic systems, known as master-slave synchronization. Although these types of observers already exist, they require accurate knowledge of the nonlinear dynamics of the system. In order to overcome that, the author introduced a model-free SMO, based on a proportional correction of the sign function of the measurement of the synchronization error. As an example, the author applied the SMO to the Lorenz system, a nonlinear system which, when tuned to certain gains, exhibits chaotic behavior.

In addition to exhibiting outstanding tracking in quadrotor applications, as outlined in the previous section, SMC has also been used in controlling underwater vehicles as the following papers outline. Aerial and underwater environments are alike, in that each presents heightened dynamics and significant disturbances to vehicles, which is why SMC is a suitable control system for controlling vehicles in such environments.

Salgado-Jimenez et al. [12] studied the performance of a model-free sliding-proportional-derivative (Sliding-PD) controller on an underwater robot, compared to a typical PID controller.
and a model-based sliding mode controller. The underwater system was considered a 1 degree-of-freedom (DOF) system, since it was physically restricted to moving only in the z direction by design. All 3 control laws were derived by the authors, and the PID and Sliding-PD controllers were tuned to the desired gain values and performance. The experiments were conducted to compare between the derived control laws, where in each case, the system was required to track a desired sinusoidal wave for 10 seconds, and a triangular wave for 10 seconds as well. When comparing performance, the controllers achieved similar tracking responses, with the proposed model-free Sliding-PD controller displaying the least mean square error in both cases.

Raygosa-Barahona et al. [13] also developed a model-free style SMC system for an underactuated underwater robot, by introducing a model-free backstepping technique with integral SMC. Since a typical two-step backstepping controller requires exact knowledge of the system model and parameters, the proposed methodology proved to be very powerful. The authors derived the control design from a PID controller, which needed to be tuned in order to achieve the desired performance, and developed the model-free backstepping technique. After establishing the required parameters, the authors performed a simulation of the model, and the vehicle converged to the desired trajectory without any chattering.

Munoz-Vazquez et al. [14] reformulated a model-free integral SMC system to quadrotor control design, by introducing the method of control to passive velocity field (VF) navigation of quadrotors. The VF was used to establish the desired path for the quadrotor in a certain environment where obstacles might be present. A sliding surface was used to force the states onto the desired trajectory. The sliding surface was designed by the authors without a dynamic model of the system, which ensured stability against parameter uncertainty. However, a VF was used in the design of the sliding surface to track the desired trajectory. The model was simulated
twice, in an environment without any obstacles, and one with obstacles to prove the usefulness of the VF to navigate around obstacles in cluttered environments. The system displayed robust tracking in both cases, without any chattering in the control effort, or states.

Mizov and Crassidis [15] introduced a novel approach to a model-free pure sliding mode control scheme for stabilizing uncertain linear and nonlinear systems. The proposed controller relied only on state measurements, which are usually available on most systems through sensor measurements or state observers, previous control input, which is also readily accessible, and knowledge of the order of the system. In order to eliminate chattering, a boundary layer, which will be described in a later section, was applied to the control law, which successfully smoothed the control effort, but reduced tracking precision. The controller was simulated on a linear and a nonlinear mass-spring-damper system. In both systems, near perfect tracking was achieved and asymptotic stability of the closed-loop system was observed.

Reis and Crassidis [16] derived a similar model-free SMC system to that proposed in [15], but utilized a distinct approach, producing a more precise controller, while maintaining the same requirements of system knowledge. The work extended the application into systems with non-unitary control input gains, which require a different approach in the design of a SMC law. The presence of measurement noise from sensors, due to the instrument's inaccuracy and outside disturbances, was also studied. The authors first simulated the controller on a nonlinear mass-spring-damper system with non-unitary control input gain, and without the presence of equipment and sensor noise. The second simulation was performed using the same system model, but included state measurement noise, using a Gaussian distribution of noise, with the variance, mean, and probability distribution obtained from the sensor's datasheet. In both cases,
outstanding tracking was achieved, and chattering was eliminated by utilizing a boundary layer in the control law.

1.3 Gaps in Previous Research on Model-free SMC

In Section 1.2.1, the focus was on developing a SMC scheme, with the use of a dynamic model describing the behavior of the system to be controlled. The proposed research in this thesis will center on the design of model-free SMC schemes, which proved to be a more powerful method, especially when it comes to controlling systems which exhibit complex dynamics.

Section 1.2.2 considered work done on model-free controllers. Martinez-Guerrera et al. [11] developed a model-free SMO, which requires an observer to proportionally correct for the error. Salgado-Jimenez et al. [12], and Raygosa-Barahona et al. [13] both developed a model-free control scheme to drive underwater robots to their desired trajectory, however, their work combines SMC with certain aspects of PID control, which limits the full potential of SMC.

1.4 Research Goals

The main goal of the work herein is to extend the work done by Crassidis and Reis in [16] on model-free SMC from Single-Input-Single-Output (SISO) applications, to being able to handle Multi-Input-Multi-Output (MIMO) systems, both fully-actuated and underactuated, with unitary and non-unitary control input gains. The control system is based solely on the order of the system, system state measurements, previous control inputs, and the bounds and shape of the control input gain matrix. Additionally, the effects of actuator dynamics will be investigated in SISO and MIMO systems, in order to determine the applicability of the developed control law on
physical systems. The proposed control law will then be simulated on various systems, such as both fully-actuated and underactuated nonlinear 2 mass-spring-damper systems, and a quadrotor model.
2  **Fundamentals of the Lyapunov Theory**

One of the most important questions in control theory is whether a system is stable or not. Unstable systems are typically not useful, and potentially dangerous, which is why the aim in most cases is to control systems in a stable manner, or towards stability if inherently unstable. The ultimate goal is closed-loop stability. Classic examples to illustrate stability concepts involve systems with pendulums. A realistic pendulum, with pivot friction, is a stable system since it will always return to its equilibrium point when disturbed. A controller can also be applied on the pendulum, in the form of an actuator at the pivot point, in order to obtain a certain behavior. On the other hand, an inverted pendulum is an obvious example of an inherently unstable system, since it will always tend to fall, unless precisely positioned at its only equilibrium point. In this situation, a control system is required to stabilize the pendulum in response to disturbances.

The most popular tool to analyze system stability is the Lyapunov stability theory, introduced by mathematician Alexandr Lyapunov in [17], which included two methods for stability analysis; the linearization method, and the direct method. The former involves linearizing a nonlinear system around an operation point, and analyzing the stability of the system at that point. The direct method, which will be further discussed in a later section, utilizes the concept of the energy of a system to determine stability. Sliding mode control relies on the direct method in order to ensure the stability of the designed control system.
2.1 Nonlinear Systems and Equilibrium Points

A nonlinear dynamic system has the following form:

$$x^n_p = f_p(x; u_m; t)$$  \hspace{1cm} (2.1)

where $x_p$ is a $p \times 1$ output vector, $n$ is the order of the system, $u_m$ is the control input, $f_p(x; u_m; t)$ is an output, input, and time dependent $p \times 1$ nonlinear vector function, and $t$ is time. $p$ and $m$ are the number of outputs and inputs in a system, respectively.

The control input can also be output and time dependent:

$$u_m = g(x; t)$$  \hspace{1cm} (2.2)

A special class of nonlinear systems is linear systems. The function in this case is linearly dependent on the states and input, and is of the following form:

$$\dot{x} = [A]\dot{x} + [B]\ddot{u}$$  \hspace{1cm} (2.3)

where $\dot{x}$ is a vector of the system’s states, $[A]$ is an $n \times n$ state matrix, and $[B]$ is an $n \times m$ matrix.

2.1.1 Autonomous and Non-autonomous systems

According to [1], a nonlinear system is said to be autonomous, if it does not explicitly depend on time, and can therefore be written as:

$$x^n_p = f_p(x; u_m)$$  \hspace{1cm} (2.4)

The same property of autonomous systems applies to linear systems, which are known as Linear Time-Invariant systems (LTI). System behavior that is dependent on time is known as non-autonomous, or time-variant. The fundamental difference between autonomous and non-autonomous systems is that the state trajectory of an autonomous system is independent of the initial time. Realistically, all systems are time-variant, or non-autonomous, since system
properties do change with time. However, system parameters, in most cases, do not vary quickly over time, and therefore, the autonomous assumption is valid and assumed in this work.

### 2.1.2 Equilibrium Points

When a system’s state trajectory converges to a single point, such a point is known as an equilibrium point. State trajectories will remain on the equilibrium point as time approaches infinity. The solution to:

$$0 = f_p(x_{p_e})$$

(2.5)

produces the vector or equilibrium states of the system. Linear systems typically contain a single equilibrium point at the origin of the statespace; however, if matrix $[A]$ is singular, they could contain an infinite number of equilibrium points in the null-space of $[A]$. Nonlinear systems can have several or infinite equilibrium points. Reconsider the example of the pendulum, which has the following nonlinear equation of motion:

$$MR^2 \ddot{\theta} + b \dot{\theta} + MgR \sin \theta = 0$$

(2.6)

where $M$ is the mass of the pendulum, $R$ is the length of the pendulum, $\theta$ is the angle between the pendulum and the vertical, $b$ is the friction at the pivot point, and $g$ is the acceleration of gravity. The pendulum is graphically represented in Figure 2.1.

![Figure 2.1: A graphical representation of the classic pendulum problem.](image)
Setting $x_1 = \theta$, $x_2 = \dot{\theta}$, the state-space equation of the system becomes:

$$\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -\frac{b}{MR^2} x_2 - \frac{g}{R} \sin x_1
\end{align*}$$  \hspace{1cm} (2.7)

Eq. (2.7) clearly shows the equilibrium points to be at:

$$\begin{align*}
x_2 &= 0 \\
\sin x_1 &= 0
\end{align*}$$  \hspace{1cm} (2.8)

where $x_1 = n\pi$ produces an infinite number of equilibrium points for the pendulum.

### 2.2 Concepts of Stability

According to Slotine and Li in [1], an equilibrium state is said to be stable if, given a spherical region with radius $R > 0$, there exists $r > 0$, such that if $\|x(0)\| < r$, then $\|x(t)\| < R$ for all $t \geq 0$. Otherwise, the equilibrium point is unstable. This is referred to as Lyapunov stability. However, in most engineering applications, Lyapunov stability is not sufficient or strong enough a concept, since remaining “near” an equilibrium point is an ambiguous concept.

An equilibrium point $x_e$ is said to be asymptotically stable if it is stable, as defined above, and if in addition there exists some $r > 0$ such that $\|x(0)\| < r$ implies that $x(t) \to x_e$ as $t \to \infty$. In other words, starting at an initial point near the equilibrium point, the system trajectories will converge to the equilibrium point. However, if there exists a point $\|x_2\| < R$, and $x(t) \to x_2$ as $t \to \infty$, then the point is said to be marginally stable. Otherwise, the point is unstable. Figure 2.2 below illustrates that concept with the equilibrium point at the origin.
In other engineering applications, it is still not satisfactory to know that a system will converge to the equilibrium point 0 in infinite time, but there is also a need to estimate how fast the system trajectories will approach $x_e$. An equilibrium point is said to be exponentially stable if there exists two strictly positive numbers $\alpha$ and $\delta$ such that for $t > 0$, $\|x(t)\| \leq \alpha \|x(0)\| e^{-\delta t}$, inside of $S_R$. In other words, the state vector is converging to the equilibrium point faster than the exponential function.

One final note concerning stability; if asymptotic or exponential stability holds for any initial state, then the equilibrium point is said to be globally asymptotically or exponentially stable.

### 2.3 Lyapunov’s Direct Method

The basic essence of Lyapunov’s direct method is an application of a simple physical observation. If the total energy of a system (be it mechanical or electrical), is continuously dissipated, then the system, however complex or nonlinear, must eventually settle at an equilibrium point. Slotine and Li [1] use a simple example to illustrate this concept: a nonlinear mass-spring-damper system with the following dynamic equation:
where $m$ is the mass, $x$ is the state of the system, $c$ is the damping coefficient, $k$ is the spring constant, and $b$ is the spring stiffening coefficient. The system is shown in Figure 2.3.

Assume the mass is deflected a large distance away from its equilibrium position; it becomes difficult to predict whether the system’s behavior will be stable, since there is no general solution to Eq. (2.9), and the equation cannot be linearized because the initial condition of the state is outside the linear range. However, by examining the total energy of the system, a sum of the kinetic and potential energy, the system’s behavior can be analyzed. The total energy is defined as:

$$V(x) = \frac{1}{2} m\dot{x}^2 + \int_0^x (kx + bx^3)dx = \frac{1}{2} m\dot{x}^2 + \frac{1}{2} kx^2 + \frac{1}{4} bx^4$$  \hspace{1cm}  (2.10)$$

By inspection of the above equation, it is clear that at $x = \dot{x} = 0$, the systems total energy converges to 0, and therefore that point is asymptotically stable. Additionally, it can be shown that the system’s instability is related to the infinite growth of the mechanical energy. Furthermore, the stability properties of the system can be characterized by the change in mechanical energy of the system. By differentiating Eq. (2.10) and using Eq. (2.9), the rate of energy variation can be obtained as:
\( \dot{V}(x) = m \ddot{x} \ddot{x} + (kx + bx^3) \dot{x} = -c|\dot{x}|^3 \)  

(2.11)

Eq. (2.11) clearly implies that the energy of the system is continuously dissipated by the damper, which makes sense from a physical standpoint, until the mass finally settles down at the natural length of the spring and damper \( x = 0 \).

Lyapunov’s direct method can be extended to more complex systems by generating a scalar energy function for the system, using the dynamic nonlinear differential equation describing the system’s behavior.

2.4 Positive Definite Functions

The energy function has a couple of properties that need to be considered. First, the function \( V(x) \) must be strictly positive unless both \( x \) and \( \dot{x} \) are 0. Secondly, the derivative of the energy function, \( \dot{V}(x) \) is monotonically decreasing with \( x \) and \( \dot{x} \). A function \( V(x) \) is said to be positive definite if \( V(x) > 0 \) for any \( x \neq 0 \). \( \dot{V}(x) \) is said to be negative semi-definite if \( \dot{V}(x) \leq 0 \). Given these conditions, an equilibrium point at 0 is said to be stable. As discussed in Section 2.2, in most applications, merely stating that a system is stable is not a sufficient condition. Therefore, it is imperative to examine the requirements of asymptotic stability.

An equilibrium point at the origin is said to be asymptotically stable if \( V(x) \) is strictly positive definite and \( \dot{V}(x) \) is strictly negative definite. In this case, the system trajectory is continuously approaching the equilibrium point.

Since the above definitions apply only in the local analysis of stability, in order to expand them into the global sense, an additional condition on \( V(x) \) is necessary: \( V(x) \) must be radially unbounded. In other words, \( V(x) \to \infty \) as \( ||x|| \to \infty \). The reason for this radial unboundedness condition is to assure that the contour surface \( V(x) = V_\alpha \) corresponds to a closed curve. If the
curve is not closed, the state trajectory could possibly drift away from the equilibrium point.

Figure 2.4 illustrates that concept.

![Figure 2.4: Lyapunov surfaces illustrating the reasoning behind the radial unboundedness condition [1].](image)

Therefore, an equilibrium point at $x = 0$ is globally asymptotically stable in the Lyapunov sense if: $V(0) = 0$, $V(x) > 0$ for any $x$, $\dot{V}(x) < 0$ for any $x$, and $V(x)$ is radially unbounded.
3 Sliding Mode Control

In this chapter, the classic sliding mode control method is presented and a derivation of the control law as it applies to MIMO systems is shown. Modeling uncertainties are a common problem in control theory, especially when it comes to designing nonlinear control systems. Model imprecision is a result of uncertainty of the plant model, or from simplifications made during the formulation of the system’s dynamics. There are two major methods to handle uncertainty: One is robust control, which includes a nominal part, such as a feedback linearization or inverse control law, and an additional term to handle modeling uncertainties. Sliding mode control is a type of robust control. The second control methodology is known as adaptive control, which is similar in structure to robust control, but in addition, the model parameters are estimated and updated in real-time based on system operation.

3.1 Derivation of the Sliding Mode Control Methodology for MIMO Systems

As mentioned in the introduction, the root of the SMC method lies in transforming higher-order linear or nonlinear systems, to a 1st order system, which usually tend to be easier to control. The method works by breaking down the control problem into a reaching phase and a sliding phase. The control law forces the system states to the sliding surface, during the reaching phase, and the problem then transforms into keeping the states on the sliding surface, as they slide to equilibrium [1], as illustrated in Figure 3.1.
Consider the following MIMO system:

\[ x_p^n = f_p(x) + [B]_{p \times m}u_m \]  

(3.1)

where \( p \) and \( m \) are the number of outputs and inputs, respectively, \( x \) is the output of the system, \( f(x) \) is a linear or nonlinear uncertain continuous function of \( x \), \([B]\) is a \( p \times m \) matrix of control input gains, which could also be uncertain, but must be bounded and of a known sign, and \( u \) is the control input.

To achieve tracking, a condition is placed on the initial condition of the desired state:

\[ x_{dp}(0) = x_p(0) \]  

(3.2)

since the system states can't instantly “jump” to the desired value, Eq (3.2) guarantees tracking without a transient. In physical systems, this condition is inherently satisfied, since state measurements need to be "zeroed out" at initiation.

The time-varying sliding surface in the state-space \( R^{(n)} \) is defined as:

\[ s_m = (\frac{d}{dt} + \lambda_m)^{n-1}\dot{x}_p \]  

(3.3)
where $\lambda_m$ is a vector of positive constants, and the vector $\bar{x}_p = x_p - x_{d,p}$ defines the tracking error.

Henceforth, the tracking problem is simplified to the equivalent of remaining on the surface $s$ for all $t > 0$.

The control law forcing the scalar quantity $s$ to 0 is to be derived using the following inequality, known as the sliding condition:

$$\frac{1}{2} \frac{d}{dt} s_m^2 \leq -\eta_m |s_m|$$  \hspace{1cm} (3.4)

where $\eta_m$ is a vector of small positive constants.

By satisfying Eq. (3.4), asymptotic stability is guaranteed, since the equation also satisfies Lyapunov's stability criteria, as described in Lyapunov's direct method. Once on the sliding surface, the controller forces the states to remain on the surface, and slide towards the origin.

The robustness of the SMC law to modeling imprecision and outside disturbances is achieved through the introduction of a discontinuous term in the control law, producing the final form of the control input:

$$u_m = \hat{u}_m - K_m \text{sgn}(s_m)$$  \hspace{1cm} (3.5)

where $K_m$ is the switching gain, which ensures asymptotic stability of the closed-loop system during the reaching portion of the control scheme, and $\text{sgn}(s_m)$ is the signum function defined as:

$$\begin{align*}
\text{sgn}(s) &= 1, \text{if } s > 0 \\
\text{sgn}(s) &= -1, \text{if } s < 0
\end{align*}$$  \hspace{1cm} (3.6)
3.2 Defining the Boundary Layer

Since the value of $s$ is never known with infinite precision, and instantaneous switching is not possible in practice, the system states tend to chatter along the sliding surface, as shown in Figure 3.2, which leads to an undesirable increase in control activity, and high frequency dynamics, which can lead to the excitation of unmodeled dynamics and damage to the physical components of the system.

![Figure 3.2: A graphical representation of chattering along the sliding surface.](image)

In order to overcome chattering, a thin boundary layer is introduced, as shown below, around the sliding surface, which acts as a low-pass filter structure to the local dynamics of the sliding surface, eliminating high frequency activity of the control law due to the switching variable.

![Figure 3.3: The boundary layer introduced neighboring the sliding surface. $\varphi$ is the thickness of the boundary layer, and $\varepsilon$ is the boundary layer width.](image)
The width of the boundary layer $\varepsilon$, is given by:

$$\varepsilon = \frac{\varphi}{\lambda^{n-1}}$$

(3.7)

where $\varphi$ is the boundary layer thickness.

Eq. (3.4) is updated accordingly, to assert the attractiveness of the boundary layer, thereby guaranteeing the distance to the boundary layer is always decreasing, resulting in the following:

$$|s_m| \geq \varphi_m \rightarrow \frac{1}{2} \frac{d}{dt} s_m^2 \leq (\varphi_m - \eta_m)|s_m|$$

(3.8)

which guarantees that the distance to the boundary layer is always decreasing.

In order to satisfy Eq. (3.8), a new switching gain given by:

$$\bar{K}_m = K_m - \dot{\varphi}_m$$

(3.9)

is generated and used in the new control law:

$$\hat{u}_m = u_m - \bar{K}_m \text{sat}(\frac{s_m}{\varphi_m})$$

(3.10)

where $\text{sat}(\frac{s_m}{\varphi_m})$ is the saturation function, defined as:

$$\text{sat}(\frac{s}{\varphi}) = \begin{cases} 
\frac{s}{\varphi} & \text{if } |\frac{s}{\varphi}| \leq 1 \\
\text{sgn}(\frac{s}{\varphi}) & \text{otherwise}
\end{cases}$$

(3.11)

Finally, the time-varying boundary layer thickness is given by the following differential equation:

$$\dot{\varphi}_m + \lambda_m \varphi_m = K_m(x_d^p)$$

(3.12)

where $\varphi_m(0) = \frac{\eta_m}{\lambda_m}$.

The above boundary layer method will produce a smooth response, which will protect the system against high frequency dynamics, but at the cost of guaranteed tracking to the precision variable $\varepsilon$, as oppose to “perfect” tracking [1].

22
3.3 Illustrative Example

An uncertain nonlinear MIMO system of 1st-order equations will be used here to illustrate the sliding mode control method. This will also aid in comparing the classic method to the proposed model-free method, and their respective results.

Consider the following system:

\[
\dot{\mathbf{x}} = \begin{bmatrix}
-2\alpha_1 x_1 + \alpha_2 x_2^2 \\
-4\alpha_1 \sin(x_2) + 5x_1 + \alpha_2 x_1^5
\end{bmatrix} + [B] \ddot{u}
\] (3.13)

where \(\alpha_1\) and \(\alpha_2\) are uncertain time-varying parameters with the following known bounds:

\[
\forall t \geq 0, 1 < \alpha_1 < 3, 2 < \alpha_2 < 4
\] (3.14)

The control input gain is \([B] = [1,1]^T\), and the reference trajectories to be tracked are:

\[
x_{1d}(t) = \sin\left(\frac{\pi}{2} t\right)
\]
\[
x_{2d}(t) = \sin(t)
\] (3.15)

Using Eq. (3.3), the sliding surface for this system is:

\[
\mathbf{s} = \mathbf{x} + \lambda \int \mathbf{x} \, dt
\] (3.16)

where \(\lambda\) in this case is the same constant for both outputs. To ensure the states remain on the sliding surface once they reach it, the derivative of the sliding surface is set to 0:

\[
\dot{s} = \dot{x} + \lambda \ddot{x} = 0
\] (3.17)

Expanding the tracking error (\(\dot{x}\)) term and substituting Eq. (3.13):

\[
\dot{s} = \begin{bmatrix}
-2\alpha_1 x_1 + \alpha_2 x_2^2 \\
-4\alpha_1 \sin(x_2) + 5x_1 + \alpha_2 x_1^5
\end{bmatrix} - \dot{x}_d + [B] \ddot{u} + \lambda \ddot{x} = 0
\] (3.18)

Solving for the control input and substituting unitary control input gain the following is obtained:
In order to handle the uncertainties present in the system, a discontinuous term is introduced to the control law, as described in Eq. (3.5):

\[
\ddot{u} = \left[1\right]\left(-\left[-2\alpha_1 x_1 + \alpha_2 x_2^2 + 4\alpha_1 \sin(x_2) + 5x_1 + \alpha_2 x_1^5\right] + \dot{x}_d - \lambda \ddot{x}\right)
\] (3.19)

To ensure the system will achieve asymptotic stability during the reaching phase, Lyapunov’s direct method, as described in section 2.3, will be applied, using the following Lyapunov function:

\[
\dot{V}(\ddot{x}) = \frac{1}{2} \ddot{s}^2
\] (3.21)

which is clearly positive definite and radially unbounded. Differentiating Eq. (3.21) produces:

\[
\dot{V}(\ddot{x}) = \ddot{s} \dddot{s}
\] (3.22)

Substituting Eq. (3.18) into Eq. (3.22), and setting the result to be strictly negative to ensure global asymptotic stability:

\[
\dot{V}(\ddot{x}) = \ddot{s} \left(\left[-2\alpha_1 x_1 + \alpha_2 x_2^2 + 4\alpha_1 \sin(x_2) + 5x_1 + \alpha_2 x_1^5\right] - \dot{x}_d + [B] \ddot{u} + \lambda \ddot{x}\right) \leq 0
\] (3.23)

Finally, substituting the control law in Eq. (3.20) into Eq. (3.23):

\[
\dot{V}(\ddot{x}) = \ddot{s} \left(\left[[\ddot{F}] - \dot{x}_d + [B]^{-1}[[-[\ddot{F}] + \dot{x}_d - \lambda \ddot{x} - \eta \text{sgn}(\ddot{s})]] + \lambda \ddot{x}\right) \leq 0
\] (3.24)

Simplifying Eq. (3.24):

\[
\dot{V}(\ddot{x}) \leq -\eta |s|
\] (3.25)

which proves that asymptotic stability will be maintained using the control law from Eq. (3.20).

The switching gain \(\ddot{K}\) replaces \(\eta\) in Eq. (3.20), and the control law estimate is written as:
\[
\hat{u} = [\hat{B}]^{-1} \left[ -\left[ -2\hat{\alpha}_1 x_1 + \hat{\alpha}_2 x_2^2 \right] + \hat{x}_d - \lambda \hat{x} - \bar{K} \text{sgn}(\hat{s}) \right]
\]

(3.26)

where the estimated parameters as generated using the known bounds as:

\[
\hat{\alpha}_1 = \sqrt{1 \times 3} = \sqrt{3} \\
\hat{\alpha}_2 = \sqrt{2 \times 4} = \sqrt{8}
\]

(3.27)

Since \([B]\) is assumed to be unitary, then \([\hat{B}] = [1,1]'\) as well. The only step remaining is to calculate the switching gain \(\bar{K}\). To do so, Eq. (3.25) is used:

\[
\tilde{s} \left( [\hat{F}] - \hat{x}_d + \left[ \begin{array}{c} 1 \\ 1 \end{array} \right] \left( [\hat{F}] + \hat{x}_d - \lambda \hat{x} - \bar{K} \text{sgn}(\hat{s}) \right) \right) \leq -\eta |s|
\]

(3.28)

which can be rewritten as:

\[
\tilde{s} \left( -[\hat{F}] + [\hat{F}] + \left[ \begin{array}{c} 1 \\ 1 \end{array} \right] \hat{x}_d - \hat{x}_d - \left[ \begin{array}{c} 1 \\ 1 \end{array} \right] \lambda \hat{x} + \lambda \hat{x} - [B][\hat{B}]^{-1} \bar{K} \text{sgn}(\hat{s}) \right) \leq -\eta |s|
\]

(3.29)

Simplifying further and solving for the switching gain:

\[
\bar{K} |\hat{s}| \geq \tilde{s} \left( -[\hat{F}] + \left[ \begin{array}{c} 1 \\ 1 \end{array} \right] \hat{x}_d - \hat{x}_d - \left[ \begin{array}{c} 1 \\ 1 \end{array} \right] \lambda \hat{x} + \left[ \begin{array}{c} 1 \\ 1 \end{array} \right] \lambda \hat{x} \right) + \left[ \begin{array}{c} 1 \\ 1 \end{array} \right] \eta \hat{s}
\]

(3.30)

Rewriting the result produces the following:

\[
\bar{K} |\hat{s}| \geq \tilde{s} \left( \left[ \begin{array}{c} 1 \\ 1 \end{array} \right] \left( [\hat{F}] - [\hat{F}] \right) \right) + \left[ \begin{array}{c} 1 \\ 1 \end{array} \right] \eta \hat{s}
\]

(3.31)

To ensure that the switching gain is conservatively greater than Eq. (3.30), the equation will be set equal to the right-hand side with an absolute value:

\[
\bar{K} = \left| \left[ \begin{array}{c} 1 \\ 1 \end{array} \right] \left( [\hat{F}] - [\hat{F}] \right) \right| + \left[ \begin{array}{c} 1 \\ 1 \end{array} \right] \eta
\]

(3.32)

The controller parameters used were \(\lambda = 20\) and \(\eta = 0.1\). The system and controller were modeled using Matlab and Simulink, with the ode5 (Dormand-Prince) solver and a step size of 0.001 seconds. The following figures display the results of the simulation:
Figure 3.4: **Left**: A comparison of output \( x_1(t) \) and the desired trajectory \( x_{1d}(t) \). **Right**: The tracking error, \( x_1 - x_{1d} \), shows miniscule error, in the order of \( 10^{-3} \).

Figure 3.5: **Left**: A comparison of output \( x_2(t) \) and the desired trajectory \( x_{2d}(t) \). **Right**: The tracking error, \( x_2 - x_{2d} \), shows miniscule error, in the order of \( 10^{-3} \).

Figure 3.6: **Left**: A comparison of the 1\textsuperscript{st} derivative of output \( x_1(t) \) and the desired trajectory \( \dot{x}_{1d}(t) \). **Right**: The 1\textsuperscript{st} derivative tracking error, \( \dot{x}_1 - \dot{x}_{1d} \), shows considerable error, in the order of \( 10^0 \).
Figure 3.7: **Left:** A comparison of the 1\textsuperscript{st} derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$. **Right:** The 1\textsuperscript{st} derivative tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows considerable error, in the order of $10^0$.

Figure 3.8: **Left:** The controller effort $u_1$. **Right:** The controller effort $u_2$.

Figure 3.9: **Left:** The sliding surface condition of controller 1. **Right:** The sliding surface condition of controller 2.
Figures 3.4 and 3.5 display perfect tracking (the term perfect tracking is common in sliding mode control applications to describe the best possible tracking while maintaining asymptotic stability) of both outputs, with minimal error, which is expected since the system under control is fully-actuated. However, the systems highest-order states display poor tracking, as shown in Figures 3.6 and 3.7, as well as high frequency chattering. The latter issue is also evident in both control inputs, in Figure 3.8. The sliding condition is satisfied for both controllers, as shown in Figure 3.9. In order to handle the issue of chattering, a boundary layer, as defined in section 3.2, is derived for the control system.

A new control law for the system is defined:

\[ \hat{u} = [\hat{B}]^{-1} \left( -2\hat{a}_1 x_1 + \hat{a}_2 x_2^2 \right) + \hat{x}_d - \lambda \hat{x} - (\hat{K} - \hat{\varphi}) \text{sat} \left( \frac{\hat{s}}{\hat{\varphi}} \right) \]  \hspace{1cm} (3.33)

where \( \hat{\varphi} \) is as defined in Eq. (3.12).

The simulation is repeated with the updated control law employing the boundary layer, and Figures 3.10 to 3.15 display the results:

**Figure 3.10:** Left: A comparison of output \( x_1(t) \) and the desired trajectory \( x_{1d}(t) \). Right: The tracking error, \( x_1 - x_{1d} \), shows miniscule error, in the order of \( 10^{-2} \).
Figure 3.11: **Left:** A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. **Right:** The tracking error, $x_2 - x_{2d}$, shows miniscule error, in the order of $10^{-2}$.

Figure 3.12: **Left:** A comparison of the 1st derivative of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$. **Right:** The 1st derivative tracking error, $\dot{x}_1 - \dot{x}_{1d}$, shows minimal error, in the order of $10^{-2}$.

Figure 3.13: **Left:** A comparison of the 1st derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$. **Right:** The 1st derivative tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows minimal error, in the order of $10^{-2}$. 
With the inclusion of the boundary around the sliding surface, all of the system’s states, Figures 3.10, 3.11, 3.12, and 3.13 all display nearly perfect tracking, without any chattering. However, although the system outputs $x_1$ and $x_2$ display adequate tracking, the tracking error of both outputs is noticeably larger compared to the previous results, due to the inclusion of the boundary layer. Adverse chattering of the control efforts has been eliminated, as shown in Figure 3.14 (compared to Figure 3.8), and the newly defined boundary layer condition, Eq. (3.8), is satisfied for both controllers, as shown in Figure 3.15. The next chapter will introduce the model-free SMC system for MIMO systems, both fully-actuated and underactuated. The 3rd Section will revisit the above example and compare the results.
4 The Model-free SMC Scheme for MIMO systems

Before starting the process of deriving the model-free SMC control law for MIMO systems, a definition of the different shapes of MIMO systems should be considered. As demonstrated in the previous section’s illustration of the classic SMC scheme, the control input gain matrix \([B]\) is required to be invertible for the control law to exist. However, the inversion of the input gain matrix is only achievable in fully-actuated MIMO systems, also known as square systems. Further manipulation of underactuated, or non-square systems, is required in order to proceed with the derivation of the model-free control law. Section 1 of this chapter will define square and non-square MIMO systems. Section 2 will derive the proposed model-free method for fully-actuated systems, followed by the derivation for non-square systems in the 3rd Section. Each section will contain examples illustrating the effectiveness of the control law on a series of 1st-order and 2nd-order systems. The last section will show simulation results of utilizing the control law on a quadrotor model.

4.1 The Definition of Square and Non-square MIMO Systems

Consider the following \(n^{th}\)-order autonomous MIMO system:

\[
x_p^n = f_p(x) + [B]_{p \times m}u_m
\]  

(4.1)

where \(p\) and \(m\) are the number of outputs and inputs, respectively, \(x\) is the system states and output, \(f(x)\) defines the autonomous nonlinear character in \(x\), \(B\) is a \(p \times m\) matrix of control input gains, and \(u\) is the control input.

The nature of a system, whether square or non-square, is based on the dimensions of matrix \([B]\). A system with as many inputs as outputs \((m = p)\) is said to be square, with
dimensions $m \times m$, whose columns are linearly independent. This implies that each output of the system has its own controller, and “perfect” control can be achieved on all outputs simultaneously, as illustrated in the Section 3.3.

On the other hand, a non-square system can be one with more inputs than outputs ($m > p$), which is considered overactuated. In this case, the matrix $[B]$ is said to be "wide", which means there is an abundance of control inputs, or superfluous inputs. Certain systems exist in such form due to the nature of their application, where safety is of great concern. In this manner, a failure of one controller can be replaced by a backup.

However, a majority of systems in practice are underactuated non-square systems. In this case, the number of inputs is less than the number of outputs ($m < p$), and the matrix $[B]$ is said to be "tall", with a lack of control inputs. As shown in the literature review in Section 1.2, previous research studied the application of the SMC method on various underactuated dynamical systems, including a quadrotor.

Knowledge of the "shape" of the control input gain matrix $[B]$ is essential to the formulation of the model-free SMC scheme, since the existence of the inverse $[B]^{-1}$ is necessary. Although systems considered here are nonlinear, the characteristics of the matrix $[B]$, as well as the terminology introduced, apply to both linear and nonlinear systems.

### 4.2 Model-free SMC for Square MIMO Systems

In this section, a model-free SMC system for square MIMO systems is developed. The derivation is similar to that performed by Reis and Crassidis in [16], since the system is fully-actuated. The only characteristics of the system required to be known are the order of the system, and an estimate of the control input gain bounds.
4.2.1 System Description

Consider the following \( n \)-th order autonomous system:

\[
x_p^n = f_p(x) + [B]_{m \times m} u_m
\]  
(4.2)

where \( p \) and \( m \) are the number of outputs and inputs, respectively, \( x \) is the system states and output, \( f(x) \) defines the autonomous nonlinear character in \( x \), \( B \) is a square \( m \times m \) matrix of control input gains, and \( u \) is the control input.

The system is redefined in the following form:

\[
x_p^n = x_p^n + [B]u_m - [B]u_{m_{k-1}} - [B]u_m + [B]u_{m_{k-1}}
\]  
(4.3)

where \( u_{m_{k-1}} \) is the previous control input. Note that Eq. (4.3) is identity in nature.

The elements of the control input gain \([B]\) are considered to be unknown, but with known bounds, as defined in the following equation:

\[
b_{lower} \leq b_{mm} \leq b_{upper}
\]  
(4.4)

where \( b_{mm} \) is an element of the control input gain matrix \([B]\).

An error parameter, \( \epsilon \), describing the error between the current control input \( u_m \) and the previous control input \( u_{m_{k-1}} \) is defined as:

\[
\epsilon_m = u_{m_{k-1}} - u_m
\]  
(4.5)

In order to compute the control law without encountering an algebraic loop throughout the simulation, an estimate of the control input error is defined as:

\[
\hat{\epsilon}_m = u_{m_{k-1}} - u_{m_{k-2}}
\]  
(4.6)

where \( u_{m_{k-2}} \) is the previous control input of the previous control input. The control input error, although not exactly known, is assumed to be bounded by the following inequality:

\[
(1 - \sigma_l)\hat{\epsilon}_m \leq \epsilon_m \leq (1 + \sigma_u)\hat{\epsilon}_m
\]  
(4.7)
where \( \sigma_l \) and \( \sigma_u \) are the lower and upper bounds, respectively, of the control input error estimate. At high sampling times, the error estimate will equal the actual error, thus the bounds will be approximately zero.

### 4.3 First-order Square MIMO System Control Law

The sliding surface for a 1\textsuperscript{st}-order system, using Eq. (3.3) is:

\[
\ddot{s} = \ddot{x} + \lambda \int \ddot{x} \, dt \tag{4.8}
\]

where \( \lambda \) in this case is the same constant for both outputs. To ensure the states remain on the sliding surface once reaching the surface, the derivative of the sliding surface is set to 0:

\[
\dot{s} = \ddot{s} = \ddot{x} + \dot{\lambda} \ddot{x} = 0 \tag{4.9}
\]

Substituting Eq. (4.3) for a 1\textsuperscript{st}-order system into Eq. (4.8):

\[
\dot{s} = \dot{x} + [B]\ddot{u} - [B]\ddot{u}_{k-1} + [B]\dot{\varepsilon} - \dot{x}_d + \lambda \ddot{x} = 0 \tag{4.10}
\]

Solving for the control input \( \ddot{u} \) and introducing a discontinuous term to ensure against uncertainties [1], results in:

\[
\ddot{u} = -[B]^{-1} \left[ \dot{x} - \dot{x}_d + \dot{\lambda} \ddot{x} + \eta \text{sgn}(s) \right] + \ddot{u}_{k-1} - \dot{\varepsilon} \tag{4.11}
\]

### 4.3.1 Asymptotic Stability of the Controller

To ensure the system will achieve asymptotic stability during the reaching phase, Lyapunov’s direct method, as described in Section 2.3, will be applied using the following Lyapunov function:

\[
\bar{V}(\bar{x}) = \frac{1}{2} \bar{s}^2 \tag{4.12}
\]

which is a positive definite and radially unbounded. Differentiating Eq. (4.12) yields:
\[ \dot{V}(\vec{x}) = \ddot{s} \dddot{s} \quad (4.13) \]

Substituting Eq. (4.10) into Eq. (4.13), and setting the result to be strictly negative to ensure global asymptotic stability results in:

\[ \dot{V}(\vec{x}) = \ddot{s}(\vec{x} + [B]\vec{u} - [B]\vec{u}_{k-1} + [B]\vec{e} - \vec{x}_d + \lambda \vec{x}) \leq 0 \quad (4.14) \]

Substituting the control law from Eq. (4.11):

\[ \dot{V}(\vec{x}) = \ddot{s}(\vec{x} + [B](-[B]^{-1}[\vec{x} - \vec{x}_d + \lambda \vec{x} + \eta sgn(\ddot{s})] + \vec{u}_{k-1} - \vec{e}) - [B]\vec{u}_{k-1} + [B]\vec{e} - \vec{x}_d + \lambda \vec{x}) \leq 0 \quad (4.15) \]

Simplifying the result:

\[ \dot{V}(\vec{x}) = \ddot{s}(-\eta sgn(\ddot{s})) \leq 0 \quad (4.16) \]

The signum function is negative unitary when the sliding surface is negative, and positive unitary when the sliding surface is positive, \( \ddot{s}(sgn(\ddot{s})) \) can be replaced with \( |\ddot{s}| \) which yields:

\[ \dot{V}(\vec{x}) = -\eta |\ddot{s}| \leq 0 \quad (4.17) \]

which is always satisfied since \( \eta \) is strictly positive. Therefore, the derivative of the Lyapunov function is negative definite and the closed-loop system is asymptotically stable.

**4.3.2 The Switching Gain**

The control law from Eq. (4.11) can be updated to include the switching gain:

\[ \vec{u} = [\vec{\hat{B}}]^{-1}[-\vec{\dot{x}} + \vec{x}_d - \lambda \vec{x} - \vec{K} sgn(\vec{s})] + \vec{u}_{k-1} - \vec{\dot{e}} \quad (4.18) \]

where \( \vec{K} \), the switching gain, ensures the state trajectories are asymptotically stable during the reaching phase. \( [\vec{\hat{B}}] \) is a matrix of the estimated control input gains for each control input. Each estimate is calculated using the following geometric mean equation.
\[
\hat{b} = \sqrt{b_{\text{upper}} b_{\text{lower}}} \quad (4.19)
\]

where \(b_{\text{upper}}\) and \(b_{\text{lower}}\) are the upper and lower bounds of the control input gain, respectively.

In order to further simplify several equations at a later stage of the derivation, an auxiliary variable is defined as (derivation detailed in [1]):

\[
\beta = \hat{b} b^{-1} = \frac{b_{\text{upper}}}{\sqrt{b_{\text{lower}}}} \quad (4.20)
\]

The above equations, Eq. (4.19) and Eq. (4.20), are applied to each element in the control input gain matrix \([B]\) to produce the estimates and auxiliary variables.

Using the sliding condition, defined in Eq. (3.4):

\[
\dot{s}s \leq -\eta|s| \quad (4.21)
\]

asymptotic stability of the closed-loop system is ensured. Eq. (4.21) is used in the derivation of the switching gain. Substituting in for \(\dot{s}\) using Eq. (4.10) with the control law from Eq. (4.18) yields:

\[
\ddot{s} \left( (\ddot{x} - \dot{x}_d) \left( 1 - [B][\dot{B}]^{-1} \right) + \lambda \ddot{x} \left( 1 - [B][\dot{B}]^{-1} \right) - [B][\dot{B}]^{-1} K \text{sgn}(s) + [B](\ddot{e} - \dot{e}) \right) \\
\leq -\eta|s| 
\]

(4.22)

The upper bound of the error estimate from Eq. (4.7) is used to ensure the resulting control law is conservative, therefore:

\[
\ddot{s} \left( (\ddot{x} - \dot{x}_d) \left( 1 - [B][\dot{B}]^{-1} \right) + \lambda \ddot{x} \left( 1 - [B][\dot{B}]^{-1} \right) - [B][\dot{B}]^{-1} K \text{sgn}(s) + [B]\sigma_u \dot{e} \right) \\
\leq -\eta|s| 
\]

(4.23)

Solving Eq. (4.23) for \(K\), and using the auxiliary variable from Eq. (4.20):

\[
\ddot{s} \left( (\ddot{x} - \dot{x}_d)([\beta] - 1) + \lambda \ddot{x}([\beta] - 1) + [\dot{B}]\sigma_u \dot{e} \right) + [\beta] \eta|s| \leq K|s| 
\]

(4.24)
In order to ensure the controller is robust to the most extreme cases of uncertainty, the inequality in Eq. (4.24), an absolute value is applied to the left-hand side and the inequality is set to equal. The following is obtained for the switching gain:

\[
\vec{K} = |\dot{x} - \dot{x}_d||[\beta] - 1| + \lambda |\ddot{x}||[\beta] - 1| + |[\vec{B}]\sigma_u \hat{\delta}| + [\beta]\eta
\]  \tag{4.25}

Substituting in for the error estimate using Eq. (4.6), the final forms of the switching gain and the control input are:

\[
\vec{K} = |\dot{x} - \dot{x}_d||[\beta] - 1| + \lambda |\ddot{x}||[\beta] - 1| + |[\vec{B}]\sigma_u (\bar{u}_{k-2} - \bar{u}_{k-1})| + [\beta]\eta
\]  \tag{4.26}

\[
\bar{u} = [\vec{B}]^{-1}[-\dot{x} + \dot{x}_d - \lambda \ddot{x} - K\text{sgn}(s)] + 2\bar{u}_{k-1} - \bar{u}_{k-2}
\]  \tag{4.27}

### 4.3.3 The Boundary Layer

The addition of the discontinuous term into the control input, as observed in Eq. (4.27), causes high frequency chattering of the control effort, which is not physically feasible in real systems and can cause damage to actuators and motors. Therefore, a smoothing boundary layer is added into the formulation of the control input as such:

\[
\bar{u} = [\vec{B}]^{-1}[-\dot{x} + \dot{x}_d - \lambda \ddot{x} - (\vec{K} - \hat{\phi})\text{sat}\left(\frac{\vec{s}}{\hat{\phi}}\right)] + 2\bar{u}_{k-1} - \bar{u}_{k-2}
\]  \tag{4.28}

where the boundary layer dynamics, according to Slotine and Li in [1], are defined as:

\[
\dot{\hat{\phi}} + \lambda \hat{\phi} = \vec{K}(\ddot{x}_d)
\]  \tag{4.29}

which are essentially the dynamics of a 1st-order filter, with \( \hat{\phi}(0) = \frac{\eta}{\lambda} \).

### 4.3.4 Illustrative Example of System of First-order Equations

The system used to simulate the developed model-free controller is the same system used in Section 3.3:
\[ \dot{x} = \begin{bmatrix} -2 \alpha_1 x_1 + \alpha_2 x_2^2 \\ -4 \alpha_1 \sin(x_2) + 5x_1 + \alpha_2 x_1^3 \end{bmatrix} + [B] \ddot{u} \quad (4.30) \]

where \( \alpha_1 \) and \( \alpha_2 \) are uncertain time-varying parameters with the following known bounds:

\[ \forall t \geq 0, |\alpha_1| \leq 1, |\alpha_2| \leq 2 \quad (4.31) \]

The control input gain is \([B] = [1,1]'\), and the reference trajectories to be tracked are:

\[
x_{1d}(t) = \sin\left(\frac{\pi}{2} t\right) \\
x_{2d}(t) = \sin(t) \quad (4.32)
\]

The simulation was performed using \( \sigma_u = 0.5, \lambda = 20, \eta = 0.1 \), on Simulink, with the fixed-step solver \text{ode5} (Dormand-Prince) at a sampling time of 0.0001, for 20 seconds. The following figures display the results of the simulation:

Figure 4.1: Left: A comparison of output \( x_1(t) \) and the desired trajectory \( x_{1d}(t) \). Right: The tracking error, \( x_1 - x_{1d} \), shows miniscule error, in the order of \( 10^{-5} \).
Figure 4.2: Left: A comparison of output $x_2(t)$ and the desired trajectory $x_{2,d}(t)$. Right: The tracking error, $x_2 - x_{2,d}$, shows miniscule error, in the order of $10^{-7}$.

Figure 4.3: Left: A comparison of the 1st derivative of output $x_1(t)$ and the desired trajectory $\dot{x}_{1,d}(t)$. Right: The 1st derivative tracking error, $\dot{x}_1 - \dot{x}_{1,d}$, shows miniscule error, in the order of $10^{-5}$, following the initial spike.

Figure 4.4: Left: A comparison of the 1st derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2,d}(t)$. Right: The 1st derivative tracking error, $\dot{x}_2 - \dot{x}_{2,d}$, shows miniscule error, in the order of $10^{-5}$, following the initial spike.
Figure 4.5: Left: The controller effort $u_1$. Right: The controller effort $u_2$.

Figure 4.6: The boundary layers of both controllers’ sliding surfaces.

The model-free SMC model was applied to the same system as in the example of Section 3.3. As shown in the left-hand side of Figures 4.1, 4.2, 4.3, and 4.4, perfect tracking is achieved on both outputs, with negligible tracking error. The inclusion of the boundary layer in the control input eliminated high frequency activity in the controller efforts as shown in Figure 4.5, while not having any effects on the performance of the controller, unlike in the case of the classical SMC method, Figures 3.10-13. The only issue observed is a spike at start-up in the highest-order states, as shown in the right-hand side of Figures 4.3 and 4.4. The spike is due to the inability to initialize the highest-order states in simulation. However, in physical systems, since sensors are typically initialized at startup, all state measurements are hence initialized, and this issue will not be encountered. Finally, the sliding condition from Eq. (3.4) is satisfied as shown in Figure 4.6.

40
4.4 Second-order Square MIMO System Control Law

The sliding surface for a 2nd-order system, using Eq. (3.3) is:

\[ \ddot{s} = \dot{x} + \lambda \dot{x} \]  \hspace{1cm} (4.33)

where \( \lambda \) in this case is the same constant for both outputs. To ensure the states remain on the sliding surface once reaching the surface, the derivative of the sliding surface is set to 0:

\[ \dot{s} = \ddot{x} + \lambda \dot{x} = 0 \]  \hspace{1cm} (4.34)

Substituting Eq. (4.3) for a 2nd-order system into Eq. (4.34):

\[ \dot{s} = \ddot{x} + [B] \ddot{u} - [B] \ddot{u}_{k-1} + [B] \dddot{e} - \ddot{x}_d + \lambda \dot{x} = 0 \]  \hspace{1cm} (4.35)

Solving for the control input \( \ddot{u} \) and introducing a discontinuous term to ensure against uncertainties, results in:

\[ \ddot{u} = -[B]^{-1} \left( \dddot{x} - \dddot{x}_d + \lambda \dot{x} + \eta sgn(\ddot{s}) \right) + \ddot{u}_{k-1} - \dddot{e} \]  \hspace{1cm} (4.36)

4.4.1 Asymptotic Stability of the Controller

To ensure the system will achieve asymptotic stability during the reaching phase, Lyapunov’s direct method, as described in Section 2.3, will be applied using the following Lyapunov function:

\[ \bar{V}(\dddot{x}) = \frac{1}{2} \dddot{x}^2 \]  \hspace{1cm} (4.37)

which is a positive definite and radially unbounded. Differentiating Eq. (4.37) yields:

\[ \dddot{V}(\dddot{x}) = \dddot{s} \dddot{s} \]  \hspace{1cm} (4.38)

Substituting Eq. (4.35) into Eq. (4.38), and setting the result to be strictly negative to ensure global asymptotic stability results in:
\[ \dot{V}(\ddot{x}) = \ddot{s}(\ddot{x} + [B]\ddot{u} - [B]\ddot{u}_{k-1} + [B]\ddot{e} - \dddot{x}_d + \lambda \dddot{x}) \leq 0 \]  

(4.39)

Substituting the control law from Eq. (4.36):

\[ \dot{V}(\ddot{x}) = \ddot{s}(\ddot{x} + [B](-[B]^{-1}\left(\dddot{x} - \dddot{x}_d + \eta \text{sgn}(\dddot{s})\right) + \dddot{u}_{k-1} - \dddot{e}) - [B]\dddot{u}_{k-1} + [B]\dddot{e} - \dddot{x}_d + \lambda \dddot{x}) \leq 0 \]  

(4.40)

Simplifying the result:

\[ \dot{V}(\ddot{x}) = \ddot{s}(-\eta \text{sgn}(\dddot{s})) \leq 0 \]  

(4.41)

The signum function is negative unitary when the sliding surface is negative, and positive unitary when the sliding surface is positive, \( \ddot{s}(\text{sgn}(\dddot{s})) \) can be replaced with \( |\ddot{s}| \) which yields:

\[ \dot{V}(\ddot{x}) = -\eta |\ddot{s}| \leq 0 \]  

(4.42)

which is always satisfied since \( \eta \) is strictly positive. Therefore, the derivative of the Lyapunov function is negative definite and the closed-loop system is asymptotically stable.

### 4.4.2 The Switching Gain

The control law from Eq. (4.36) can be updated to include the switching gain:

\[ \ddot{u} = [\dddot{B}]^{-1}(\dddot{x} - \dddot{x}_d - \lambda \dddot{x} - \dddot{K} \text{sgn}(\dddot{s}) + \dddot{u}_{k-1} - \dddot{e}) \]  

(4.43)

where \( \dddot{K} \) ensures the state trajectories are asymptotically stable during the reaching phase. \([\dddot{B}]\) is a matrix of the estimated of control input gains for each control input.

Using the sliding condition, defined in Eq. (3.4):

\[ \dddot{s} \ddot{s} \leq -\eta |\ddot{s}| \]  

(4.44)
asymptotic stability of the closed-loop system is ensured. Eq. (4.21) is used in the derivation of the switching gain. Substituting in for \( \dot{s} \) using Eq. (4.35) with the control law from Eq. (4.36) yields:

\[
\dot{s} \left( \ddot{x} - \ddot{x}_d \right) \left( 1 - [B][\hat{B}]^{-1} \right) + \lambda \dot{s} \left( 1 - [B][\hat{B}]^{-1} \right) - [B][\hat{B}]^{-1} \dot{K} sgn(s) + [B](\varepsilon - \hat{\varepsilon}) \\
\leq -\eta |s|
\]

The upper bound of the error estimate from Eq. (4.7) is used to ensure the resulting control law is conservative:

\[
\dot{s} \left( \ddot{x} - \ddot{x}_d \right) \left( 1 - [B][\hat{B}]^{-1} \right) + \lambda \dot{s} \left( 1 - [B][\hat{B}]^{-1} \right) - [B][\hat{B}]^{-1} \dot{K} sgn(s) + [B]\sigma_u \hat{\varepsilon} \\
\leq -\eta |s|
\]

Solving Eq. (4.46) for \( \dot{K} \), and using the auxiliary variable from Eq. (4.20):

\[
\dot{s} \left( (\ddot{x} - \ddot{x}_d)([\beta] - 1) + \lambda \dot{s}(\beta) - 1 \right) + [\beta] \eta |s| \leq \dot{K} |s|
\]

In order to ensure the controller can handle the most extreme case of the inequality in Eq. (4.47), an absolute value is applied to the left-hand side and the inequality is set to equal. The following is obtained for the switching gain:

\[
\dot{K} = |\ddot{x} - \ddot{x}_d| |[\beta] - 1| + \lambda |\dot{s}| |[\beta] - 1| + |[\beta]\sigma_u \hat{\varepsilon}| + [\beta] \eta
\]

Substituting in for the error estimate using Eq. (4.6), the final forms of the switching gain and the control input are:

\[
\dot{K} = |\ddot{x} - \ddot{x}_d| |[\beta] - 1| + \lambda |\dot{s}| |[\beta] - 1| + |[\beta]\sigma_u (u_{k-2} - u_{k-1})| + [\beta] \eta \\
\dot{u} = [\hat{B}]^{-1} \left( -\ddot{x} + \ddot{x}_d - \lambda \ddot{s} - \dot{K} sgn(s) \right) + 2u_{k-1} - u_{k-2}
\]
4.4.3 The Boundary Layer

The addition of the discontinuous term into the control input, as observed in Eq. (4.50), causes high frequency chattering of the control effort, which is not physically feasible in real systems and can cause damage to actuators and motors. Therefore, a smoothing boundary layer is added into the formulation of the control input as such:

$$
\ddot{u} = [\hat{B}]^{-1} \left( -\ddot{x} + \dot{x}_d - \lambda \ddot{x} - (\bar{K} - \bar{\phi}) \text{sat} \left( \frac{\ddot{\bar{\phi}}}{\phibar} \right) \right) + 2\ddot{u}_{k-1} - \ddot{u}_{k-2}
$$

(4.51)

where the boundary layer dynamics, according to Slotine and Li in [1], are defined as:

$$
\ddot{\bar{\phi}} + \lambda \bar{\phi} = \bar{K}(\ddot{x}_d)
$$

(4.52)

which are essentially the dynamics of a 1st-order filter, with $\bar{\phi}(0) = \frac{\eta}{\lambda}$.

4.4.4 Illustrative Example of System of Second-order Equations

The system used to simulate the developed model-free controller is a fully-actuated nonlinear mass-spring-damper, shown in the figure below:

![Figure 4.7: A fully-actuated 2 mass-spring-damper system.](image)

The equations of motion of the system in Figure 4.7 are as follows:
\[
\begin{align*}
\dot{m}_1 \dot{x}_1 &= u_1 + c_2 (\dot{x}_2 - \dot{x}_1) \dot{x}_2 - \dot{x}_1| + k_2 (x_2 - x_1) \\
- \beta_2 (x_2 - x_1)^3 - c_1 \dot{x}_1 \dot{x}_1| - k_1 x_1 + \beta_1 x_1^3 \\
\dot{m}_2 \dot{x}_2 &= u_2 - c_2 (\dot{x}_2 - \dot{x}_1) \dot{x}_2 - \dot{x}_1| - k_2 (x_2 - x_1) \\
+ \beta_2 (x_2 - x_1)^3 
\end{align*}
\]

(4.53)

where \( m_i \) is the mass, \( x_i \) is the output, \( u_i \) is the input, \( c_i \) is the damping coefficient, given as \( \vec{c} = [5,8]' \), \( k_i \) is the spring constant, given as \( \vec{k} = [3,7]' \), \( \beta_i \) is the spring stiffening coefficient, given as \( \vec{\beta} = [-1.5, -3]' \), and \( i \) is the number of masses, springs and dampers.

After dividing through by the mass, the control input gain becomes \( \frac{1}{m_1}, \frac{1}{m_2} \)'. The masses are assumed to be uncertain, but with known bounds:

\[
\begin{align*}
5 &\leq m_1 \leq 15 \\
15 &\leq m_2 \leq 25 
\end{align*}
\]

(4.54)

The reference signals to be tracked are:

\[
\begin{align*}
x_{1d}(t) &= \sin \left( \frac{\pi}{2} t \right) \\
x_{2d}(t) &= \sin \left( \frac{\pi}{2} t \right) 
\end{align*}
\]

(4.55)

The simulation was performed using the same parameters as in the previous example of the system of \( 1^{st} \)-order equations. The following figures display the results of the simulation:

**Figure 4.8:** Left: A comparison of output \( x_1(t) \) and the desired trajectory \( x_{1d}(t) \). Right: The tracking error, \( x_1 - x_{1d} \), shows miniscule error, in the order of \( 10^{-6} \).
Figure 4.9: **Left:** A comparison of output $x_2(t)$ and the desired trajectory $x_{2,d}(t)$. **Right:** The tracking error, $x_2 - x_{2,d}$, shows miniscule error, in the order of $10^{-6}$.

Figure 4.10: **Left:** A comparison of the velocity of output $\dot{x}_1(t)$ and the desired trajectory $\dot{x}_{1,d}(t)$. **Right:** The velocity tracking error, $\dot{x}_1 - \dot{x}_{1,d}$, shows miniscule error, in the order of $10^{-6}$.

Figure 4.11: **Left:** A comparison of the velocity of output $\dot{x}_2(t)$ and the desired trajectory $\dot{x}_{2,d}(t)$. **Right:** The velocity tracking error, $\dot{x}_2 - \dot{x}_{2,d}$, shows miniscule error, in the order of $10^{-6}$.
Figure 4.12: Left: A comparison of the acceleration of output \( x_1(t) \) and the desired trajectory \( \dot{x}_{1,d}(t) \). Right: The acceleration tracking error, \( \ddot{x}_1 - \ddot{x}_{1,d} \), shows miniscule error, in the order of \( 10^{-4} \), following the initial spike.

Figure 4.13: Left: A comparison of the acceleration of output \( x_2(t) \) and the desired trajectory \( \dot{x}_{2,d}(t) \). Right: The acceleration tracking error, \( \ddot{x}_2 - \ddot{x}_{2,d} \), shows miniscule error, in the order of \( 10^{-4} \), following the initial spike.

Figure 4.14: Left: The controller effort \( u_1 \). Right: The controller effort \( u_2 \).
As shown in the left-hand side of figures 4.8 through 4.13, perfect tracking is achieved on all states of both outputs, with negligible tracking error. The controller was also able to overcome the uncertainty in the mass, and with the inclusion of the boundary layer, no chattering was observed, as shown in Figure 4.14. The issue of an initial spike in the error of the highest-order states, as shown in the right-hand side of Figures 4.12 and 4.13, is also observed in this example. Finally, the boundary layer condition is satisfied, as shown in Figure 4.15.

Perfect tracking was achievable on both masses simultaneously, since each had its dedicated control input. However, as will be seen in the next section, with fewer control inputs than outputs, a weighting function will be introduced to allow for the choice of preferred output tracking.

### 4.4 Model-free SMC for Non-square MIMO systems

This section introduces the derivation process of the model-free SMC controller for underactuated (non-square) MIMO systems. The issue arising in the case of a "tall" $[B]$ matrix is that the $[B]$ matrix is not invertible, and hence the control law $u$, as shown in Eq. (4.50) for
instance, cannot be formulated. A possible solution to this problem is to apply a coordinate transformation on the system, and by doing so, essentially "squaring" the matrix \([B]\).

### 4.4.1 System Description

Consider the following \(n^{th}\)-order autonomous system:

\[
x_p^n = f_p(x) + [B]_{p \times m}u_m
\]  
(4.56)

where \(m < p\), and the matrix \([B]\) is non-square. Let:

\[
\ddot{y} = [T]\ddot{x}
\]  
(4.57)

where the dimensions of matrix \([T]\) = the dimensions of \([B]'\). Eq. (4.56) can be rewritten as:

\[
y_p^{(n)} = [T]_{m \times p}f_p(x) + [[T]_{m \times p}[B]_{p \times m}]u_m
\]  
(4.58)

and the product of \([[T][B]]\) is now square and invertible.

The matrix \([T]\) can be thought of as a weighing matrix. Since the system in question is underactuated, and states cannot display perfect tracking simultaneously, \([T]\) can be used to track certain outputs "more heavily" than others.

To apply the model-free SMC method to an underactuated MIMO system, knowledge of the size of the \([B]\) matrix of the system is required in order to formulate the transformation matrix \([T]\). Once that is acquired, the model-free SMC scheme is developed in the \(y\) coordinate system, in a similar manner to the derivation in square MIMO systems, and \([T]\) is used to relate \(y\) to \(x\), and vice versa. The next section will formulate the model-free control law for a system of 1\textsuperscript{st}-order underactuated models.
4.5 First-order Non-Square MIMO System Control Law

To begin the derivation process, consider the following nonlinear system:

\[ \dot{x} = \bar{F}(x) + [B]u \]  

(4.59)

where \( \bar{x} \) is a \( p \times 1 \) vector of outputs, \( \bar{F} \) is a \( p \times 1 \) vector of functions of the output \( x \), \( [B] \) is a \( p \times m \) matrix, with \( m < p \), and \( u \) is a \( m \times 1 \) vector of control inputs. Using Eq. (4.57), the following is obtained:

\[ \dot{y} = [T]\bar{F}(x) + [[T][B]]u \]  

(4.60)

The transformation matrix is applied to all states, the functions describing the system, the control input and its gain, and the desired trajectories as well, before they are used in deriving the control law. The formulation process is then done in the \( y \) domain.

Updating Eq. (4.8):

\[ \ddot{s}_y = \dot{y} + \lambda \int \dot{y} \, dt \]  

(4.61)

Differentiating and equating to \( \ddot{0} \):

\[ \dot{s}_y = \dot{y} + \lambda \ddot{y} = \ddot{0} \]  

(4.62)

Substituting in Eq. (4.3) for a 1st-order sliding surface, but in the \( y \) domain:

\[ \dot{s}_y = \dot{y} + \bar{[T][B]}u_y - \bar{[T][B]}u_{y_{k-1}} + \bar{[T][B]}\ddot{\bar{y}} + \lambda \ddot{y} - \dot{y}_d = 0 \]  

(4.63)

where \( u_y \) is the control input in the \( y \) domain. \( u \) is extracted from \( u_y \) through the the transformation matrix \( [T] \) as follows:

\[ u = [T]^{-1}u_y \]  

(4.64)

Solving for the control input and introducing the discontinuous term:
\[ \ddot{u}_y = -\left([T][B]\right)^{-1} \left( \dot{\tilde{y}} - \dot{y}_d + \lambda \tilde{y} - \eta \text{sgn}(\tilde{s}_y) \right) + \ddot{u}_{y_{k-1}} - \tilde{e}_y \]  

(4.65)

### 4.5.1 Asymptotic Stability of the Controller

To ensure the system will achieve asymptotic stability during the reaching phase, Lyapunov’s direct method, as described in Section 2.3, will be applied, using the following Lyapunov function:

\[ \tilde{V}(\tilde{y}) = \frac{1}{2} \tilde{s}_y^2 \]  

(4.66)

which is a positive definite and radially unbounded. Differentiating Eq. (4.66) yields:

\[ \dot{\tilde{V}}(\tilde{y}) = \dot{s}_y \tilde{s}_y \]  

(4.67)

Substituting Eq. (4.63) into Eq. (4.67), and setting the result to be strictly negative to ensure global asymptotic stability results in:

\[ \dot{\tilde{V}}(\tilde{y}) = \dot{s}_y \left( \dot{\tilde{y}} + [T][B] \ddot{u}_y - [T][B] \ddot{u}_{y_{k-1}} + [T][B] \tilde{e}_y + \lambda \tilde{y} - \dot{y}_d \right) \leq 0 \]  

(4.68)

Substituting the control law from Eq. (4.65):

\[ \dot{\tilde{V}}(\tilde{y}) = \dot{s}_y \left( \dot{\tilde{y}} + [T][B] \left( -\left([T][B]\right)^{-1} \left( \dot{\tilde{y}} - \dot{y}_d + \lambda \tilde{y} - \eta \text{sgn}(\tilde{s}_y) \right) + \ddot{u}_{y_{k-1}} - \tilde{e}_y \right) \right) \right) \]  

(4.69)

\[ \]  

Simplifying the result:

\[ \dot{\tilde{V}}(\tilde{y}) = \dot{s}_y \left( -\eta \text{sgn}(\tilde{s}_y) \right) \leq 0 \]  

(4.70)

The signum function is negative unitary when the sliding surface is negative, and positive unitary when the sliding surface is positive, \( \tilde{s}_y \left( \text{sgn}(\tilde{s}_y) \right) \) can be replaced with \( |\tilde{s}_y| \) which produces:
which is always satisfied since $\eta$ is strictly positive. Therefore, the derivative of the Lyapunov function is negative definite and the closed-loop system is asymptotically stable.

### 4.5.2 The Switching Gain

The control law from Eq. (4.65) can be updated to include the switching gain:

$$
\ddot{u}_y = -\left([T][\hat{B}]\right)^{-1}\left(\ddot{y} - \ddot{y}_d + \lambda\ddot{y} - \bar{K}_y sgn(\ddot{s}_y)\right) + \ddot{u}_{y,k-1} - \ddot{e}_y
$$

where $\bar{K}_y$ ensures the state trajectories are asymptotically stable during the reaching phase. $[\hat{B}]$ is a matrix of the estimated of control input gains for each control input.

In order to further simplify several equations at a later stage of the derivation, an auxiliary variable is defined as:

$$
[\beta_y] = [T][\hat{B}][T][B]^{-1}
$$

Using the sliding condition, defined in Eq. (3.4):

$$
\ddot{s}_y \dddot{s}_y \leq -\eta|\dddot{s}_y|
$$

asymptotic stability of the closed-loop system is ensured. Eq. (4.74) is used in the derivation of the switching gain. Substituting in for $\ddot{s}_y$ using Eq. (4.63) with the control law from Eq. (4.65) yields:

$$
\dddot{s}_y \left(\ddot{y} - \ddot{y}_d\right)\left(1 - [T][B]\right)\left([T][\hat{B}]\right)^{-1} + \lambda\ddot{y} \left(1 - [T][B]\right)\left([T][\hat{B}]\right)^{-1} - \left([T][B]\right)\left([T][\hat{B}]\right)^{-1} \bar{K}_y sgn(\dddot{s}_y) + [T][B]\left(\dddot{e}_y - \dddot{e}_y\right) \leq -\eta|\dddot{s}_y|
$$
The upper bound of the error estimate from Eq. (4.7) is used to ensure the resulting control law is conservative:

\[
\hat{s}_y \left( (\hat{y} - \hat{y}_d) \left( 1 - \left[ [T][B] \right] \left[ [T][\hat{B}] \right]^{-1} \right) + \lambda \hat{y} \left( 1 - \left[ [T][B] \right] \left[ [T][\hat{B}] \right]^{-1} \right) \right) - \left[ [T][B] \right] \left[ [T][\hat{B}] \right]^{-1} \bar{K}_y sgn(\hat{s}_y) + \left[ [T][B] \right] \sigma_u \hat{\xi}_y \right) \leq -\eta |\hat{s}_y|
\]

(Solving Eq. (4.76) for \(K_y\), and using the auxiliary variable from Eq. (4.73):

\[
\hat{s}_y \left( (\hat{y} - \hat{y}_d) \left( [\beta_y] - 1 \right) + \lambda \hat{y} \left( [\beta_y] - 1 \right) + \left[ [T][\hat{B}] \right] \sigma_u \hat{\xi}_y \right) + \left[ [T][B] \right] \bar{K}_y |\hat{s}_y| \leq \bar{K}_y |\hat{s}_y|
\]

In order to ensure the controller can handle the most extreme case of the inequality in Eq. (4.77), an absolute value is applied to the left-hand side and the inequality is set to equal. The following is obtained for the switching gain:

\[
\bar{K}_y = |\hat{y} - \hat{y}_d| |[\beta_y] - 1| + \lambda |\hat{y}| |[\beta_y] - 1| + \left[ [T][\hat{B}] \right] \sigma_u |\hat{\xi}_y| + [\beta_y] \eta
\]

(Substituting in for the error estimate using Eq. (4.6), the final forms of the switching gain and the control input are:

\[
\bar{K}_y = |\hat{y} - \hat{y}_d| |[\beta_y] - 1| + \lambda |\hat{y}| |[\beta_y] - 1| + \left[ [T][\hat{B}] \right] \sigma_u (\bar{u}_{y_{k-2}} - \bar{u}_{y_{k-1}}) + [\beta_y] \eta
\]

\[
\bar{u}_{y} = -\left[ [T][\hat{B}] \right]^{-1} \left( \hat{y} - \hat{y}_d + \lambda \hat{y} - \bar{K}_y sgn(\hat{s}_y) \right) + 2\bar{u}_{y_{k-1}} - \bar{u}_{y_{k-2}}
\]

4.5.3 The Boundary Layer

The addition of the discontinuous term into the control input, as observed in Eq. (4.80), causes high frequency chattering of the control effort, which is not physically feasible in real systems and can cause damage to actuators and motors. Therefore, a smoothing boundary layer is added into the formulation of the control input as such:
\[ \ddot{u}_y = -\left( [T][\beta] \right)^{-1} \left( \dot{y} - \dot{y}_d + \lambda \ddot{y} - (\bar{K}_y - \dot{\phi}_y) \text{sat} \left( \frac{\ddot{y}}{\phi_y} \right) \right) + 2 \ddot{u}_{y_{k-1}} - \ddot{u}_{y_{k-2}} \]  
(4.81)

where the boundary layer dynamics, according to Slotine and Li in [1], are defined as:

\[ \ddot{\phi}_y + \lambda \dot{\phi}_y = \bar{K}_y (\dot{y}_d) \]  
(4.82)

which are essentially the dynamics of a 1st-order filter, with \( \ddot{\phi}_y(0) = \frac{\eta}{\lambda} \).

### 4.5.4 Illustrative Example of System of Underactuated First-Order Equations

Consider the following 1-input 2-output nonlinear system of 1st-order models:

\[
\begin{align*}
\dot{x}_1 &= -2x_1 + 3 \sin(x_2) - x_1x_2 + u \\
\dot{x}_2 &= -4 \cos(x_2) + 5x_1 
\end{align*}
\]  
(4.83)

The desired trajectories are:

\[
\begin{align*}
x_{1d}(t) &= \sin \left( \frac{\pi}{2} t \right) \\
x_{2d}(t) &= \sin(t) 
\end{align*}
\]  
(4.84)

The \( B \) matrix of the system is:

\[
[B] = \begin{bmatrix} 1 \\ 0 \end{bmatrix}
\]  
(4.85)

Therefore, using the transformation matrix in Eq. (4.60), the following transformations apply to the system:

\[
\begin{align*}
\dot{\tilde{y}} &= \begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix} \dot{x}, \quad \ddot{\tilde{y}} = \begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix} \ddot{x} \\
\dot{\tilde{y}}_d &= \begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix} \dot{x}_d, \quad \ddot{\tilde{y}}_d = \begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix} \ddot{x}_d 
\end{align*}
\]

Note the above transformations do not require any previous knowledge of the mathematical model of the system. The only information required are the order of the system, and the shape of the control input matrix \([B]\).
The simulation was performed twice, starting with the $[T] = [100, 1]'$ matrix weighing $x_1$ tracking more heavily first, and then changing it to weigh $x_2$ tracking more in the second test run. Simulink was used with the same preferences as the previous example. The figures below display the results.

**Figure 4.16:** Left: A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. Right: $x_1(t)$ tracking error, $x_1 - x_{1d}$, shows miniscule error, in the order of $10^{-2}$.

**Figure 4.17:** Left: A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. Right: $x_2(t)$ tracking error, $x_2 - x_{2d}$, shows significant error, in the order of $10^0$. 
Figure 4.18: **Left:** A comparison of the 1st derivative of output $x_1(t)$ and the desired trajectory $\dot{x}_{1_d}(t)$. **Right:** The 1st derivative tracking error, $\dot{x}_1 - \dot{x}_{1_d}$, shows miniscule error, in the order of $10^{-3}$, following the initial spike.

Figure 4.19: **Left:** A comparison of the 1st derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2_d}(t)$. **Right:** The 1st derivative tracking error, $\dot{x}_2 - \dot{x}_{2_d}$, shows significant error, in the order of $10^0$.

Figure 4.20: **Left:** A verification of the sliding condition. **Right:** The controller effort.
As expected, the output $x_1(t)$ and its states did a pretty good job tracking the desired signal, as shown in Figures 4.16 and 4.18, however, $x_2(t)$ and its states did a poor job of attempting to track the desired trajectories, as shown in Figures 4.17 and 4.19, since the transformation matrix $[T]$ weighed the first output significantly more than the second output. The boundary layer, shown in the left-hand side of Figure 4.20, is satisfied.

The second run of the simulation, where $x_2(t)$ tracking will be weighed more heavily, utilized the following elements in the transformation matrix: $T_{11} = 1$; $T_{12} = 100$.

**Figure 4.21**: Left: A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. Right: $x_1(t)$ tracking error, $x_1 - x_{1d}$, shows significant error, in the order of $10^0$.

**Figure 4.22**: Left: A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. Right: $x_2(t)$ tracking error, $x_2 - x_{2d}$, shows miniscule error, in the order of $10^{-2}$. 
Figure 4.23: **Left**: A comparison of the 1\textsuperscript{st} derivative of output $x_1(t)$ and the desired trajectory $\dot{x}_{1_d}(t)$. **Right**: The 1\textsuperscript{st} derivative tracking error, $\dot{x}_1 - \dot{x}_{1_d}$, shows significant error, in the order of $10^0$, following the initial spike.

Figure 4.24: **Left**: A comparison of the 1\textsuperscript{st} derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2_d}(t)$. **Right**: The 1\textsuperscript{st} derivative tracking error, $\dot{x}_2 - \dot{x}_{2_d}$, shows miniscule error, in the order of $10^{-2}$, following the initial spike.

Figure 4.25: **Left**: A verification of the sliding condition in Eq. (5.4). **Right**: The controller effort.
In this scenario, \( x_2(t) \) outperforms \( x_1(t) \) in tracking its desired trajectory with minor negligible errors, as shown in Figure 4.22 and its 1\textsuperscript{st} derivative state tracking in Figure 4.24. The previously observed initial spike in the highest-order states, due to the inability to initialize those states, is seen here as well, in Figure 4.23 and the right-hand side of Figure 4.24. In addition, the control effort in the right-hand side of Figure 4.25 also displays an aggressive initial spike, in part due to the initialization mismatch issue, but more importantly, due to heavily weighing the un-actuated state, \( x_2(t) \), which the control input can only control through \( x_1(t) \).

### 4.6 Second-order Non-Square MIMO System Control Law

The sliding surface for a 2\textsuperscript{nd}-order system, using Eq. (3.3) is:

\[
\ddot{s}_y = \ddot{y} + \lambda \dddot{y}
\]  
(4.86)

Differentiating and equating to \( \ddot{0} \):

\[
\dot{s}_y = \dot{y} + \lambda \dddot{y} = \ddot{0}
\]  
(4.87)

Substituting in Eq. (4.3) for a 2\textsuperscript{nd}-order sliding surface, but in the \( y \) domain:

\[
\dot{s}_y = \ddot{y} + [(T)[B]] \ddot{u}_y - [(T)[B]] \ddot{u}_{y_{k-1}} + [(T)[B]] \dddot{e}_y + \lambda \dddot{y} - \dddot{y}_d = \ddot{0}
\]  
(4.88)

Solving for the control input and introducing the discontinuous term to ensure against uncertainties results in:

\[
\ddot{u}_y = -[T][B]^{-1} \left( \ddot{y} - \ddot{y}_d + \lambda \dddot{y} + \eta \text{sgn}(\dddot{s}_y) \right) + \ddot{u}_{y_{k-1}} - \ddot{e}_y
\]  
(4.89)
4.6.1 Asymptotic Stability of the Controller

To ensure the system will achieve asymptotic stability during the reaching phase, Lyapunov’s direct method, as described in section 2.3, will be applied using the following Lyapunov function:

\[ V(\hat{y}) = \frac{1}{2} \hat{s}_y^2 \]  

(4.90)

which is a positive definite and radially unbounded. Differentiating Eq. (4.90) yields:

\[ \dot{V}(\hat{y}) = \dot{s}_y \ddot{s}_y \]  

(4.91)

Substituting Eq. (4.88) into Eq. (4.91), and setting the result to be strictly negative to ensure global asymptotic stability results in:

\[ \dot{V}(\hat{y}) = \ddot{s}_y \left( \ddot{y} + [[T][B]] \ddot{u}_y - [[T][B]] \ddot{u}_{y_{k-1}} + [[T][B]] \ddot{e}_y + \lambda \ddot{y} - \ddot{y}_d \right) \leq 0 \]  

(4.92)

Substituting in the control law from Eq. (4.89):

\[ \dot{V}(\hat{y}) = \ddot{s}_y \left( \ddot{y} + [[T][B]] \left( -[[T][B]]^{-1} \left( \ddot{y} - \ddot{y}_d + \lambda \ddot{y} + \eta \text{sgn}(\tilde{s}_y) \right) + \ddot{u}_{y_{k-1}} - \ddot{e}_y \right) \right) \]  

\[ \quad - [[T][B]] \ddot{u}_{y_{k-1}} + [[T][B]] \ddot{e}_y + \lambda \ddot{y} - \ddot{y}_d \right) \leq 0 \]  

(4.93)

Simplifying the result:

\[ \dot{V}(\hat{y}) = \ddot{s}_y \left( -\eta \text{sgn}(\tilde{s}_y) \right) \leq 0 \]  

(4.94)

The signum function is negative unitary when the sliding surface is negative, and positive unitary when the sliding surface is positive, \( \tilde{s}_y \text{sgn}(\tilde{s}_y) \) can be replaced with \( |\tilde{s}_y| \) which yields:

\[ \dot{V}(\hat{y}) = -\eta |\tilde{s}_y| \leq 0 \]  

(4.95)

which is always satisfied since \( \eta \) is strictly positive. Therefore, the derivative of the Lyapunov function is negative definite and the closed-loop system is asymptotically stable.
4.6.2 The Switching Gain

The control law from Eq. (4.89) can be updated to include the switching gain:

$$\tilde{u}_y = -\left([T][\tilde{B}]\right)^{-1} (\ddot{y} - \ddot{y}_d + \lambda \dot{\tilde{y}} + \tilde{K}_y sgn(\tilde{s}_y)) + \tilde{u}_{y_{k-1}} - \tilde{e}_y$$ (4.96)

where $\tilde{K}_y$ ensures the state trajectories are asymptotically stable during the reaching phase. $[\tilde{B}]$ is a matrix of the estimated of control input gains for each control input.

Using the sliding condition, defined in Eq. (3.4):

$$\dot{s}_y \tilde{s}_y \leq -\eta |\tilde{s}_y|$$ (4.97)

asymptotic stability of the closed-loop system is ensured. Eq. (4.97) is used in the derivation of the switching gain. Substituting in for $\dot{s}_y$ using Eq. (4.88) with the control law from Eq. (4.89):

$$\tilde{s}_y \left( (\ddot{y} - \ddot{y}_d) \left( 1 - [[T][B]] \left[[T][\tilde{B}]\right]^{-1} \right) + \lambda \dot{\tilde{y}} \left( 1 - [[T][B]] \left[[T][\tilde{B}]\right]^{-1} \right) \right)$$

$$- [[T][B]] \left[[T][\tilde{B}]\right]^{-1} \tilde{K}_y sgn(\tilde{s}_y) + [[T][B]](\tilde{e}_y - \dot{\tilde{s}}_y) \right) \leq -\eta |\tilde{s}_y|$$ (4.98)

The upper bound of the error estimate from Eq. (4.7) is used to ensure the resulting control law is conservative so that:

$$\tilde{s}_y \left( (\ddot{y} - \ddot{y}_d) \left( 1 - [[T][B]] \left[[T][\tilde{B}]\right]^{-1} \right) + \lambda \dot{\tilde{y}} \left( 1 - [[T][B]] \left[[T][\tilde{B}]\right]^{-1} \right) \right)$$

$$- [[T][B]] \left[[T][\tilde{B}]\right]^{-1} \tilde{K}_y sgn(\tilde{s}_y) + [[T][B]] \sigma_u \dot{\tilde{s}}_y \right) \leq -\eta |\tilde{s}_y|$$ (4.99)

Solving Eq. (4.99) for $K_y$, and using the auxiliary variable from Eq. (4.73):

$$\tilde{s}_y \left( (\ddot{y} - \ddot{y}_d) ([\beta_y] - 1) + \lambda \dot{\tilde{y}} ([\beta_y] - 1) + [[T][\tilde{B}] \sigma_u \dot{\tilde{s}}_y \right) + [\beta_y] \eta |\tilde{s}_y| \leq K_y |\tilde{s}_y|$$ (4.100)
In order to ensure the controller can handle the most extreme case of the inequality in Eq. (4.100), an absolute value is applied to the left-hand side and the inequality is set to equal. The following is obtained for the switching gain:

\[
\vec{K}_y = |\vec{y} - \hat{\vec{y}}_d|||\beta_y| - 1| + \lambda |\hat{\vec{y}}| ||\beta_y| - 1| + ||[T][\hat{\vec{B}}]| \sigma_u |\hat{\vec{\epsilon}}_y| + [\beta_y]\eta
\]

(4.101)

Substituting in for the error estimate using Eq. (4.6), the final forms of the switching gain and the control input are:

\[
\begin{align*}
\vec{K}_y &= |\vec{y} - \hat{\vec{y}}_d|||\beta_y| - 1| + \lambda |\hat{\vec{y}}| ||\beta_y| - 1| + ||[T][\hat{\vec{B}}]| \sigma_u (\vec{u}_{y_{k-2}} - \vec{u}_{y_{k-1}}) + [\beta_y]\eta \\
\vec{u}_y &= -[T][\hat{\vec{B}}]^{-1} \left( \vec{y} - \hat{\vec{y}}_d + \lambda \hat{\vec{y}} + \vec{K}_y sgn(\vec{s}_y) \right) + 2\vec{u}_{y_{k-1}} - \vec{u}_{y_{k-2}}
\end{align*}
\]

(4.102) (4.103)

### 4.6.3 The Boundary Layer

The addition of the discontinuous term into the control input, as can be seen in Eq. (4.103), causes high frequency chattering of the control effort, which is not physically feasible in real systems and can cause damage to actuators and motors. Therefore, a smoothing boundary layer is added into the formulation of the control input as such:

\[
\begin{align*}
\vec{u}_y &= -[T][\hat{\vec{B}}]^{-1} \left( \vec{y} - \hat{\vec{y}}_d + \lambda \hat{\vec{y}} + (\vec{K}_y - \vec{\phi}_y) sat \left( \frac{\vec{s}_y}{\vec{\phi}_y} \right) \right) + 2\vec{u}_{y_{k-1}} - \vec{u}_{y_{k-2}}
\end{align*}
\]

(4.104)

where the boundary layer dynamics, according to Slotine and Li in [1], are defined as:

\[
\dot{\vec{\phi}}_y + \lambda \vec{\phi}_y = \vec{K}_y(\hat{\vec{y}}_d)
\]

(4.105)

which are essentially the dynamics of a 1st-order filter, with \( \vec{\phi}_y(0) = \frac{\eta}{\lambda} \).
4.6.4 Illustrative Example of System of Underactuated Second-Order Equations

The system considered here to illustrate the effectiveness of the model-free controller is an underactuated nonlinear mass-spring-damper, shown in the figure below:

![Figure 4.26: An underactuated 2 mass-spring-damper system.](image)

The equations of motion of the system in Figure 4.26 are as follows:

\[
\begin{align*}
\ddot{m}_1 \dot{x}_1 &= c_2 (\dot{x}_2 - \dot{x}_1) |\dot{x}_2 - \dot{x}_1| + k_2 (x_2 - x_1) \\
&\quad - \beta_2 (x_2 - x_1)^3 - c_1 \dot{x}_1 |\dot{x}_1| - k_1 x_1 + \beta_1 x_1^3 \\
\ddot{m}_2 \dot{x}_2 &= u - c_2 (\dot{x}_2 - \dot{x}_1) |\dot{x}_2 - \dot{x}_1| - k_2 (x_2 - x_1) \\
&\quad + \beta_2 (x_2 - x_1)^3
\end{align*}
\]

(4.106)

where \( m_i \) is the mass, \( x_i \) is the output, \( u \) is the only input of the system, \( c_i \) is the damping coefficient, given as \( c = [5,8]' \), \( k_i \) is the spring constant, given as \( k = [3,7]' \), \( \beta_i \) is the spring stiffening coefficient, given as \( \beta = [-1.5,-3]' \), and \( i \) is the number of masses, springs and dampers. The desired trajectories are:

\[
\begin{align*}
x_{1d}(t) &= \sin(t) \\
x_{2d}(t) &= \sin(t)
\end{align*}
\]

(4.107)

After dividing through by the mass, the control input gain becomes \([B] = [0,1/m_2]'\). The masses are assumed to be uncertain, but with known bounds:
Therefore, using the transformation matrix as shown in Eq. (4.60), the following transformations apply to the system:

\[
\begin{align*}
\vec{y} &= \begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix}' \tilde{x}, \quad \check{\vec{y}} = \begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix}' \check{x}, \quad \vec{\ddot{y}} = \begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix}' \ddot{x} \\
\vec{y}_d &= \begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix}' \tilde{x}_d, \quad \check{\vec{y}}_d = \begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix}' \check{x}_d, \quad \vec{\ddot{y}}_d = \begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix}' \ddot{x}_d
\end{align*}
\]

The simulation was preformed twice, starting with the \([T]\) matrix weighing \(x_1\) tracking more heavily first, and then changing it to weigh \(x_2\) tracking more in the second run. Simulink was used with the same preferences as previous examples. The first case utilized the following elements in the transformation matrix: \(T_{11} = 1.25; T_{12} = 0.02\). The figures below display the results. This specific transformation matrix was obtained by implementing an optimization routine in order to ensure the weights used were not too high as to cause the system to go unstable, while achieving perfect tracking in the desired state. The optimization method is outlined in Section 4.8.

**Figure 4.27**: Left: A comparison of output \(x_1(t)\) and the desired trajectory \(x_{1,d}(t)\). Right: \(x_1(t)\) tracking error, \(x_1 - x_{1,d}\), shows minimal error, in the order of \(10^{-2}\).
Figure 4.28: Left: A comparison of output $x_2(t)$ and the desired trajectory $x_{2_d}(t)$. Right: $x_2(t)$ tracking error, $x_2 - x_{2_d}$, shows considerable error, in the order of $10^{-1}$.

Figure 4.29: Left: A comparison of the velocity of output $x_1(t)$ and the desired trajectory $\dot{x}_{1_d}(t)$. Right: The velocity tracking error, $\dot{x}_1 - \dot{x}_{1_d}$, shows miniscule error, in the order of $10^{-2}$.

Figure 4.30: Left: A comparison of the velocity of output $x_2(t)$ and the desired trajectory $\dot{x}_{2_d}(t)$. Right: The velocity tracking error, $\dot{x}_2 - \dot{x}_{2_d}$, shows some error, in the order of $10^{-1}$. 

65
**Figure 4.31:** **Left:** A comparison of the acceleration of output $x_1(t)$ and the desired trajectory $\ddot{x}_{1d}(t)$. **Right:** The acceleration tracking error, $\ddot{x}_1 - \ddot{x}_{1d}$, shows miniscule error, in the order of $10^{-2}$, following the initial spike.

**Figure 4.32:** **Left:** A comparison of the acceleration of output $x_2(t)$ and the desired trajectory $\ddot{x}_{2d}(t)$. **Right:** The acceleration tracking error, $\ddot{x}_2 - \ddot{x}_{2d}$, shows significant error, in the order of $10^1$, following the initial spike.

**Figure 4.33:** **Left:** A verification of the sliding condition. **Right:** The controller effort.
Figures 4.27, 4.29, and 4.31 show perfect tracking of the desired trajectory by mass 1, which is expected with chosen \([T]\) matrix, while \(x_2\) does not track the desired signal well, as shown in Figures 4.28, 4.30, and 4.32. Again, the initial spike was observed in the highest-order state of both outputs. This should not be an issue in physical systems since all states, along with their measurements, should be at zero at initialization. The boundary layer in the left-hand side of Figure 4.33 is still satisfied.

The second run of this simulation, where \(x_2(t)\) will be weighed more heavily, utilized the following elements in the transformation matrix: \(T_{11} = 0.02; T_{12} = 1.25\).

**Figure 4.34:** Left: A comparison of output \(x_1(t)\) and the desired trajectory \(x_{1d}(t)\). Right: \(x_1(t)\) tracking error, \(x_1 - x_{1d}\), shows considerable error, in the order of \(10^1\).

**Figure 4.35:** Left: A comparison of output \(x_2(t)\) and the desired trajectory \(x_{2d}(t)\). Right: \(x_2(t)\) tracking error, \(x_2 - x_{2d}\), shows minimal error, in the order of \(10^2\).
Figure 4.36: Left: A comparison of the velocity of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$. Right: The velocity tracking error, $\ddot{x}_1 - \dot{x}_{1d}$, shows some error, in the order of $10^{-1}$.

Figure 4.37: Left: A comparison of the velocity of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$. Right: The velocity tracking error, $\ddot{x}_2 - \dot{x}_{2d}$, shows miniscule error, in the order of $10^{-2}$.

Figure 4.38: Left: A comparison of the acceleration of output $x_1(t)$ and the desired trajectory $\ddot{x}_{1d}(t)$. Right: The acceleration tracking error, $\dddot{x}_1 - \ddot{x}_{1d}$, shows significant error, in the order of $10^{-1}$.
Figure 4.39: **Left:** A comparison of the acceleration of output $x_2(t)$ and the desired trajectory $\ddot{x}_{2d}(t)$. **Right:** The acceleration tracking error, $\dddot{x}_2 - \dddot{x}_{2d}$, shows miniscule error, in the order of $10^{-2}$.

Figure 4.40: **Left:** A verification of the sliding condition. **Right:** The controller effort.

In this case, $x_2(t)$ replaces $x_1(t)$ in performance, tracking its desired trajectory with minor errors, while $x_1(t)$ displays inferior performance, as compared to the previous case. An important thing to note is that the control effort in the second simulation, shown in the right-hand side of Figure 4.40, is less aggressive than the control effort in the previous case, shown in in the right side of Figure 4.33, which was cropped to display the content of the signal in more detail. There are two reasons for this behavior: First, due to the inability to initialize the highest order state of the system in simulation, a start-up transient is typically observed in the control effort along with the highest order states. However, the second and more likely reason for the behavior...
of the control effort in the right of Figure 4.33 is the structure of the system under control. Since only 1 control input exists in the system, and it is directly applied onto mass 2, the only way the controller can drive the states of mass 1 onto the desired trajectories is through mass 2, which is not optimal, and causes the controller to experience this peculiar and aggressive behavior. The left-hand side of Figure 4.40 displays the satisfaction of the boundary layer condition.

### 4.7 Position Control of a Quadrotor

In this section, the developed MIMO model-free SMC system is simulated on a quadrotor, an underactuated system. The model-free controller was used to obtain perfect position tracking of the quadrotor in x-y-z. The mathematical model used was obtained from Xu et al. in [4], divided into two subsystems, a fully-actuated subsystem:

$$
\begin{bmatrix}
\dot{z}

\dot{\phi}
\end{bmatrix} =
\begin{bmatrix}
    u_1 \cos \phi \cos \psi \\
    u_4
\end{bmatrix} -
\begin{bmatrix}
    g \\
    -K_3 \dot{z} / m
\end{bmatrix}
$$

(4.109)

and an underactuated subsystem:

$$
\begin{bmatrix}
\dot{x} \\
\dot{y}
\end{bmatrix} =
\begin{bmatrix}
    u_1 \cos \phi & u_1 \sin \phi \\
    u_1 \sin \phi & -u_1 \cos \phi
\end{bmatrix}
\begin{bmatrix}
    \sin \theta \cos \psi \\
    \sin \psi
\end{bmatrix} +
\begin{bmatrix}
    -K_1 \dot{x} / m \\
    -K_2 \dot{y} / m
\end{bmatrix}
$$

(4.110)

$$
\begin{bmatrix}
\dot{\theta} \\
\dot{\psi}
\end{bmatrix} =
\begin{bmatrix}
    u_2 \\
    u_3
\end{bmatrix} +
\begin{bmatrix}
    -K_4 \dot{\theta} / l_1 \\
    -K_5 \dot{\psi} / l_2
\end{bmatrix}
$$

where \((x, y, z)\) are the position coordinates, \((\theta, \psi, \phi)\) are the three Euler angles, representing pitch, roll, and yaw, respectively, \(K_i\)'s are drag coefficients, \(g\) is the acceleration of gravity, \(m\) is the mass of the quadrotor, \(l\) is half the length of the quadrotor, and \(I_i\)'s are the moments of inertia with respect to each axis.
\( u_i \)'s are virtual control inputs defined as:

\[
\begin{align*}
  u_1 &= (F_1 + F_2 + F_3 + F_4)/m \\
  u_2 &= l(-F_1 - F_2 + F_3 + F_4)/l_1 \\
  u_3 &= l(-F_1 + F_2 + F_3 - F_4)/l_2 \\
  u_4 &= C(F_1 - F_2 + F_3 - F_4)/l_3
\end{align*}
\]  

(4.111)

where \( F_i \)'s are the thrusts generated by each rotor, and \( C \) is a force to moment scaling factor.

The fully-actuated subsystem in Eq. (4.109), describing altitude and heading, can be controlled using the SMC model derived for square systems, and perfect tracking can be achieved on both outputs. However, perfect tracking in the underactuated subsystem in Eq. (4.110) can only be achievable on two of the four outputs since the system contains only two inputs.

Based on the dynamics of a quadrotor, pitch and roll are the driving variables of \( x \) and \( y \) positions, respectively. Therefore, the 2x4 \( [T] \) matrix used in the simulation was constructed to achieve perfect control in \( x \) and \( y \), since most applications typically involve the desire to reach a certain position in space, at a certain heading and altitude, as oppose to achieving perfect control of pitch and roll angles.

The application of the model-free SMC system is similar to the previous example in section 4.6.5, and Simulink was used with the same preferences as previous examples. The table of parameters used is shown below:

<table>
<thead>
<tr>
<th>System Parameters</th>
<th>Desired Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>( l_1, l_2 )</td>
<td>1.25 Ns^2/rad</td>
</tr>
<tr>
<td>( l_3 )</td>
<td>2.5 Ns^2/rad</td>
</tr>
<tr>
<td>( K_1, K_2, K_3 )</td>
<td>0.010 Ns/m</td>
</tr>
<tr>
<td>( K_4, K_5, K_6 )</td>
<td>0.012 Ns/rad</td>
</tr>
<tr>
<td>( m )</td>
<td>2 kg</td>
</tr>
<tr>
<td>( l )</td>
<td>0.2 m</td>
</tr>
<tr>
<td>( g )</td>
<td>9.8 m/s</td>
</tr>
</tbody>
</table>
The following results were obtained:

**Figure 4.41:** Plots of the position of the quadrotor compared to the desired trajectory.

**Figure 4.42:** **Left:** Plot of the altitude of the quadrotor. **Right:** Plot of the heading of the quadrotor.

**Figure 4.43:** **Left:** Plot of the pitch angle of the quadrotor. **Right:** Plot of the roll angle of the quadrotor.
As shown in the figures above, the controller was able to achieve perfect tracking of the desired trajectories. An important thing to note though is that since the desired trajectory is the complete path from position 0 to 1 m in both $x$ and $y$, shown in Figure 4.41, 0 to 3 m in $z$, shown in Figure 4.42 Left, and 0 to $\pi/3$ in $\varphi$, shown in Figure 4.42 Right, the controller is pretty aggressive in pitching and rolling the quadrotor, causing high frequency activity in both those terms, the price of perfect position tracking, as shown in Figures 4.44 and 4.45.
4.8 The Transformation Matrix

As seen in the previous sections, a non-square MIMO system can be controlled using the model-free SMC technique in conjunction with the use of a transformation system, which allows us to specify a weight on which outputs the controller should track more than others. This method works pretty well in achieving perfect control on the preferred outputs; however, it does involve further computation to generate the \([T]\) matrix. In single-input multi-output systems, the process is straightforward, since each element of the \([T]\) matrix directly corresponds to its respective output, as seen in sections 4.5.4 and 4.6.4. However, in larger systems, with more than one input, it becomes more difficult to correlate each element to a respective output. Therefore, an optimization routine is utilized to form the \([T]\) matrix.

4.8.1 Integral Square Error

The optimization routine used in the previous simulations is the Integral Square Error (ISE) performance index. ISE measures the performance of the system by integrating the square of the tracking error of each output over the simulation run time. A cost function based on ISE is calculated in Matlab during the simulation run, and the \([T]\) matrix is formulated as Matlab reruns the simulation in order to minimize the value of the cost function.

Consider the \([T]\) matrix of the underactuated subsystem of a quadrotor (Eq. 4.110):

\[
[T] = \begin{bmatrix}
T_{11} & T_{12} & T_{13} & T_{14} \\
T_{21} & T_{22} & T_{23} & T_{24}
\end{bmatrix}
\]  

(4.112)

Since it is unclear which element corresponds to which of the four outputs, the following ISE based cost function was constructed:
\[
J = \int_0^{t_f} (k_1 \tilde{x}^2 + k_2 \tilde{y}^2 + k_3 \tilde{\theta}^2 + k_4 \tilde{\psi}^2)dt
\]  
(4.113)

where \( t_f \) is the simulation run time, \( k_i \)'s are the weighing Scales, \( \tilde{x} = x - x_d \), \( \tilde{y} = y - y_d \), \( \tilde{\theta} = \theta - \theta_d \), and \( \tilde{\psi} = \psi - \psi_d \) are the tracking errors.

The cost function in Eq. (4.112) is used in the \textit{fminsearch} function in Matlab, which searches for a local minima by varying a variable \( x \), which in this case is the \([T]\) matrix, and observes the output of the cost function. In order to calculate the integral, the Trapezoidal numerical integration method was used to numerically approximate the integral.

Although this method is pretty effective in tuning the \([T]\) matrix to the desirable configuration, it can be computationally exhausting, especially with more complex models, thereby considerably increasing computation time.


5 Model-free SMC with Actuator Time-Delay

In real-world applications, systems are affected by an inherent time-delay due to the various actuators within the system. These include devices such as Servomotors, Pneumatic actuators, DC motors and much more. Unfortunately, computer simulations do not generally account for these delays, which results in an inaccurate representation of a system’s realistic response. Although most actuator time delays are of really short durations (in milliseconds), they still effect the performance of a system controller. Therefore, it is worthwhile to consider the effects of actuator time-delays on the performance of the developed model-free SMC methodology. This chapter will first consider the addition of an actuator time-delay into the classical SMC technique in a linear SISO system, and then the effects on the model-free SMC technique, in both linear and nonlinear, SISO and MIMO systems.

Actuator dynamics are introduced to systems in simulation using the following first-order continuous transfer function form:

\[ H(s) = \frac{1}{\tau s + 1} \]

where \( \tau \) is the actuator time constant, in seconds.

By introducing the above transfer function in between the controller output and the plant, an actuator time delay affecting the system will be simulated. Although the transfer function shown in Eq. (5.1) alters the plant model from the controller’s perspective, the SMC system design in the following examples does not take it into account. Therefore, the results display the ability of the controller to overcome a time-delay in the response, without modeling for one.
5.1 Classical SMC Scheme with Actuator Time-Delay

Consider the following 2\textsuperscript{nd} order linear mass-spring-damper system:

\[ m\ddot{x} + c\dot{x} + kx = bu \] (5.2)

where \( m, c, k, \) and \( b \) are assumed unknown but with the following known bounds:

\[
\begin{align*}
1 < m < 3 \\
0.2 < c < 1 \\
1 < k < 3 \\
0.5 < b < 1.5
\end{align*}
\] (5.3)

The reference signal to be tracked is:

\[ x_d(t) = 0.5\sin(t) \] (5.4)

The parameters used in the controller were chosen to be \( \lambda = 20, \eta = 0.1 \).

As for the actuator, the time constant in the system is assumed to be:

\[ \tau = 0.1 \text{ s} \]

The simulation was performed using Simulink, with the fixed-step solver ode5 (Dormand-Prince) at a sampling time of 0.0001, for 20 seconds. The following figures display the results of the simulation:

**Figure 5.1:** Left: A comparison of output \( x(t) \) and the desired trajectory \( x_d(t) \). Right: \( x(t) \) tracking error, \( x - x_d \), shows miniscule error, in the order of \( 10^{-4} \).
Figure 5.2: Left: A comparison of the 1st derivative of output $x(t)$ and the desired trajectory $\dot{x}_d(t)$. Right: The 1st derivative tracking error, $\dot{x} - \dot{x}_d$, begins in the order of $10^{-1}$, but then quickly drops to the order of $10^{-3}$.

Figure 5.3: Left: A verification of the boundary layer condition. Right: The controller effort.

Looking back at Figures 5.1 and 5.2, perfect tracking is observed. The effects of the actuator time delay in the system can be clearly seen in Figure 5.3. Due to the addition of the delay, there is a lag in the system between the controller output and the system response. Therefore, the controller experiences high frequency action during the initial 8 seconds, where the controller is attempting to drive the system towards the desired state but the system does not immediately respond.

The chosen actuator time constant at 0.1 seconds is a high estimate of what most actuators are rated at. Therefore, with a lower value of time delay, such as at 0.01 seconds, the
system response to the sliding mode controller is superior to what was shown in the previous example. Additionally, the control effort does not experience the undesirable high frequency action shown in Figure 5.3.

5.2 Model-free SMC Scheme for SISO Systems with Actuator Time-Delay

When considering the effects of adding the actuator transfer function to the system while utilizing a model-free sliding mode controller, a couple of things are worth noting.

First, since the controller relies on feedback from the system and previous control input in order to drive the states onto the desired trajectory, the time-delay will have an adverse effect on the controller’s function since it now takes longer for the control input to “reach” the system and feedback to the controller will be delayed. The mismatch between the previous control input and the delayed feedback is thought to be one of the issues that might affect the controller’s performance.

Additionally, the model-free technique relies on knowledge of the order of the system. Adding the actuator time-delay is in reality increasing the order of the system, since an additional pole has been introduced. With lower time constants, the pole has no severe effect on the system, but as the time-delay is increased, the pole becomes more dominant and hence has a greater effect on the plant. One solution can be to simply include the actuator transfer function in the calculations of the model-free scheme, by increasing the order of the system used to generate the control input. However, the additional state will present a challenge in the formulation of the controller since there is no desired trajectory available for the state to track.

Consider once more the linear 2\textsuperscript{nd} order mass-spring-damper system:

\[ m\ddot{x} + c\dot{x} + kx = bu \]  

(5.5)
where in this case $m$, $c$, and $k$, are assumed known as:

$$
\begin{align*}
    m &= 2 \\
    c &= 0.8 \\
    k &= 2
\end{align*}
$$

(5.6)

and $b$ is assumed to be unitary.

The reference signal to be tracked is:

$$
   x_d(t) = \sin\left(\frac{\pi}{2} t \right)
$$

(5.7)

The controller parameters were the same as in the previous example. As for the actuator, the time constant in the system is assumed to be:

$$
   \tau = 0.1 \text{s}
$$

The simulation was performed using Simulink, with the fixed-step solver ode5 (Dormand-Prince) at a sampling time of 0.0001, for 20 seconds.

Without making any changes to the model-free controller, the simulation was unstable. Therefore, some changes have to be implemented in the controller design.

Since the mismatch between the previous control input that the design relies on, and the delayed control input into the plant model, is thought to be the reason behind this error, one possible solution is to use the control input after the actuator model as the feedback into the controller design. In some cases however, such as a DC motor on a quadrotor, this is not physically possible. Therefore, a workaround is to implement the actuator transfer function within the controller design, in discrete form. The updated control input to be used in the formulation of the model-free SMC input is:

$$
\vec{u} = \hat{B}^{-1}\left(-\ddot{x} + \dot{x}_d - \lambda \dot{x} - (\hat{K} - \hat{\varphi})\text{sat}\left(\frac{\hat{s}}{\hat{\varphi}}\right)\right) + 2\hat{u}_{pa_{k-1}} - \hat{u}_{pa_{k-2}}
$$

(5.8)
where $u_{pa}$ is the control input post actuator defined as the following:

$$u_{pa} = bu_{pa_{k-1}} + au_{k-1}$$  \hspace{1cm} (5.9)$$

gains $b$ and $a$ are determined using the $c2d$ command in Matlab with the function argument being the continuous-time transfer function of the actuator model with the desired time-constant and the simulation step size. Using the updated control law results with the following:

**Figure 5.4:** **Left:** A comparison of output $x(t)$ and the desired trajectory $x_d(t)$. **Right:** $x(t)$ tracking error, $x - x_d$, shows miniscule error, in the order of $10^{-2}$.

**Figure 5.5:** **Left:** A comparison of the $1^{st}$ derivative of output $x(t)$ and the desired trajectory $\dot{x}_d(t)$. **Right:** The $1^{st}$ derivative tracking error, $\dot{x} - \dot{x}_d$, in the order of $10^{-2}$. 
Perfect tracking is observed in position and velocity, as seen in Figures 5.4 and 5.5. However, the most critical observation in the results is the sliding condition in the left-hand side of Figure 5.6, which is not satisfied in this case. The reason behind that is thought to be due to the changes implemented to the SMC design. Therefore, in order to satisfy the boundary layer condition with the updated control law, the sliding condition needs to be reformulated.

Performing the simulation with a shorter time-delay, at 0.01 seconds, results in even better tracking, which is expected due to the smaller time constant, and satisfies the boundary layer condition, as seen in the figure below:

**Figure 5.6:** Left: The boundary layer condition. Right: The controller effort.

**Figure 5.7:** A verification of the boundary layer condition.
The jump outside the boundary layer at start-up is due to the initialization problem that was seen in previous sections. In reality, physical systems will initialize at start-up.

The model-free SMC method with the actuator time-delay is simulated on a more involved example of a nonlinear 3rd order system with time-varying coefficients:

$$\ddot{x} + \alpha_1(t)\dot{x}^2 + \alpha_2(t)\dot{x}^5 \sin(4x) = b(t)u$$

(5.10)

where $\alpha_1(t)$, $\alpha_2(t)$, and $b(t)$, are uncertain time-varying functions with the following known bounds:

$$\forall t \geq 0, |\alpha_1(t)| \leq 1, |\alpha_2(t)| \leq 2, \text{ and } 1 \leq b(t) \leq 4$$

(5.11)

The reference signal to be tracked is:

$$x_d(t) = \sin\left(\frac{\pi}{2}t\right)$$

(5.12)

The controller parameters and actuator time constant were the same as in the previous example. The simulation was performed using Simulink, with the fixed-step solver ode5 (Dormand-Prince) at a sampling time of 0.0001, for 20 seconds, and included the updated control input formulation. The following figures display the results of the simulation:

![Figure 5.8: Left: A comparison of output $x(t)$ and the desired trajectory $x_d(t)$. Right: $x(t)$ tracking error, $x - x_d$, shows miniscule error, in the order of $10^{-3}$.](image)
Figure 5.9: **Left**: A comparison of the 1st derivative of output $x(t)$ and the desired trajectory $\dot{x}_d(t)$. **Right**: The 1st derivative tracking error, $\dot{x} - \dot{x}_d$ in the order of $10^{-2}$.

Figure 5.10: **Left**: A comparison of the 2nd derivative of output $x(t)$ and the desired trajectory $\ddot{x}_d(t)$. **Right**: The 2nd derivative tracking error, $\ddot{x} - \ddot{x}_d$ in the order of $10^{-1}$.

Figure 5.11: **Left**: A verification of the boundary layer condition. **Right**: The controller effort.
A couple of things to note here; first, the modified control input technique, which incorporates the actuator time-delay into the model-free SMC method, worked in this example and the boundary layer condition is perfectly satisfied, as seen in the left-hand side of Figure 5.11. This presents inconsistencies since the control law did not satisfy the boundary layer condition with a larger time delay in the 2nd order linear example of the mass-spring-damper system, but did so at 0.01 seconds. Second, it is clear that tracking performance diminishes with the increase in the order of the state, but remains reasonable. Finally, as simulation time goes on, state tracking improves, which is apparent in tracking error plots in the right-hand side of Figures 5.8, 5.9, and 5.10. This seems to suggest that the controller is able to eventually “catch up” with the system, and deliver perfect tracking as it overcomes the effects of the actuator time-delay.

5.3 Model-free SMC Scheme for Square MIMO Systems with Actuator Time-Delay

In order to avoid the inconsistencies seen in the previous section with the changing of the actuator time constant, it will be kept constant from here on out at 0.01 seconds. The first example considered here is of a two-input two-output, square, nonlinear system of 1st-order equations:

\[
\begin{align*}
\dot{x}_1 &= -2\alpha_1(t)x_1 + \alpha_2(t)x_2^2 + u_1 \\
\dot{x}_2 &= -4\alpha_1(t)\sin(x_2) + 5x_1 + \alpha_2(t)x_1^5 + u_2
\end{align*}
\]  

(5.13)

where \(\alpha_1(t)\), and \(\alpha_2(t)\) are uncertain time-varying functions with the following known bounds:

\[
\forall t \geq 0, |\alpha_1(t)| \leq 1, |\alpha_2(t)| \leq 2
\]  

(5.14)

The reference signals to be tracked are:
The controller parameters were the same as in the previous example. A simulation was performed using Simulink, with the fixed-step solver ode5 (Dormand-Prince) at a sampling time of 0.0001, for 20 seconds.

The following figures display the results of the simulation:

\[
\begin{align*}
    x_{1d}(t) &= \sin \left( \frac{\pi}{2} t \right) \\
    x_{2d}(t) &= \sin(t)
\end{align*}
\] (5.15)

Figure 5.12: **Left**: A comparison of output \(x_1(t)\) and the desired trajectory \(x_{1d}(t)\). **Right**: \(x_1(t)\) tracking error, \(x_1 - x_{1d}\), shows miniscule error, in the order of \(10^{-3}\), following the initial spike.

Figure 5.13: **Left**: A comparison of output \(x_2(t)\) and the desired trajectory \(x_{2d}(t)\). **Right**: \(x_2(t)\) tracking error, \(x_2 - x_{2d}\), shows miniscule error, in the order of \(10^{-3}\), following the initial spike.
Figure 5.14: **Left:** A comparison of the 1st derivative of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$. **Right:** The 1st derivative tracking error, $\dot{x}_1 - \dot{x}_{1d}$, shows miniscule error, in the order of $10^{-3}$, following the initial spike.

Figure 5.15: **Left:** A comparison of the 1st derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$. **Right:** The 1st derivative tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows miniscule error, in the order of $10^{-3}$, following the initial spike.

Figure 5.16: A verification of the sliding condition of the **Left:** 1st output, **Right:** 2nd output.
As seen in Figures 5.12, 5.13, 5.14, and 5.15, perfect tracking is observed in all states, with the previously encountered initialization spike in the highest-order states. At a 0.01 second actuator time delay, and the utilization the modified control input formulation, the boundary layer condition is satisfied in both outputs, as seen in Figure 5.17. An attempt was performed with the time delay at 0.1 seconds, but even though perfect tracking was observed, the boundary layer condition was not satisfied, suggesting a need for its re-formulation at higher actuator time constants, as stated earlier.

5.4 Model-free SMC Scheme for Non-square MIMO Systems with Actuator Time-Delay

The final example is of a non-square single-input two-output nonlinear system:

\[
\begin{align*}
\dot{x}_1 &= -2x_1 + 3 \sin(x_2) - x_1x_2 + u \\
\dot{x}_2 &= -4 \cos(x_2) + 5x_1
\end{align*}
\] (5.16)

The desired trajectories are:

\[
\begin{align*}
x_{1d}(t) &= \sin\left(\frac{\pi}{2} t\right) \\
x_{2d}(t) &= \sin(t)
\end{align*}
\] (5.17)
The $B$ matrix of the system is:

$$[B] = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$ \hfill (5.18)

Therefore, using the transformation matrix, the following transformations apply to the system:

\[
\begin{align*}
\ddot{\vec{y}} &= [T_{11} \\ T_{12}] \ddot{\vec{x}}, \ \dot{\vec{y}} = [T_{11} \\ T_{12}] \dot{\vec{x}} \\
\ddot{\vec{y}}_d &= [T_{11} \\ T_{12}] \ddot{\vec{x}}_d, \ \dot{\vec{y}}_d = [T_{11} \\ T_{12}] \dot{\vec{x}}_d
\end{align*}
\]

Note that the above transformations did not require any previous knowledge of the mathematical model of the system. The only information required are the order of the system, and the shape of the control input matrix $[B]$.

The simulation is performed twice, starting with the $[T]$ matrix weighing $x_1$ more heavily first, and then changing it to weigh $x_2$ more in the second test run. Simulink was used with the same preferences as previous examples. The figures below display the results.

The elements of the $[T]$ matrix in the first case are: $T_{11} = 1; \ T_{12} = 0.001$.

---

**Figure 5.18:** Left: A comparison of output $x_1(t)$ and the desired trajectory $x_{1,d}(t)$. Right: $x_1(t)$ tracking error, $x_1 - x_{1,d}$, shows miniscule error, in the order of $10^{-2}$, following the initial spike.
Figure 5.19: **Left**: A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. **Right**: $x_2(t)$ tracking error, $x_2 - x_{2d}$, shows significant error, in the order of $10^0$.

Figure 5.20: **Left**: A comparison of the $1^{st}$ derivative of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$. **Right**: The $1^{st}$ derivative tracking error, $\dot{x}_1 - \dot{x}_{1d}$, shows miniscule error, in the order of $10^2$, following the initial spike.

Figure 5.21: **Left**: A comparison of the $1^{st}$ derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$. **Right**: The $1^{st}$ derivative tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows significant error, in the order of $10^0$. 

90
As expected, the output $x_1(t)$ perfectly tracked its desired signal, as shown in Figures 5.18 and 5.20, however, $x_2(t)$ did a poor job of attempting to track its desired states, as shown in Figures 5.19 and 5.21, since the transformation matrix, $[T]$, weighed the first output significantly more than the second output. The modified controller handled the actuator time-delay pretty well, and the boundary layer condition is satisfied, as seen in the left-hand side of Figure 5.22, past the start-up transient.

In the second run of this simulation, $x_2(t)$ will be weighed more heavily, utilizing the following elements in the transformation matrix: $T_{11} = 0.1; T_{12} = 1$. 

Figure 5.23: Left: A comparison of output $x_1(t)$ and the desired trajectory $x_{1d}(t)$. Right: $x_1(t)$ tracking error, $x_1 - x_{1d}$, shows significant error, in the order of $10^0$. 

Figure 5.22: Left: A verification of the boundary layer condition. Right: The controller effort.
Figure 5.24: **Left**: A comparison of output $x_2(t)$ and the desired trajectory $x_{2d}(t)$. **Right**: $x_2(t)$ tracking error, $x_2 - x_{2d}$, shows miniscule error, in the order of $10^{-2}$, following the initial spike.

Figure 5.25: **Left**: A comparison of the 1st derivative of output $x_1(t)$ and the desired trajectory $\dot{x}_{1d}(t)$. **Right**: The 1st derivative tracking error, $\dot{x}_1 - \dot{x}_{1d}$, shows significant error, in the order of $10^0$, following the initial spike.

Figure 5.26: **Left**: A comparison of the 1st derivative of output $x_2(t)$ and the desired trajectory $\dot{x}_{2d}(t)$. **Right**: The 1st derivative tracking error, $\dot{x}_2 - \dot{x}_{2d}$, shows miniscule error, in the order of $10^{-2}$, following the initial spike.
In this case, $x_2(t)$ outperforms $x_1(t)$ in tracking the desired trajectory with minor negligible errors, as seen in Figures 5.24 and 5.26. The start-up outputs of $\dot{x}_1(t)$, and the control effort have been scaled down on the plots in the right-hand side of Figures 5.25 and 5.27 to display details of the output throughout the simulation. The actual initial values of the outputs were an order higher than what is shown. The reasoning behind these substantial values at start-up, aside from the initialization discrepancy, is the setup of the system model. Since the control input $u$ is present in the equation of $x_1(t)$, it requires more controller effort to guide $x_2(t)$ to its desired reference signal, resulting in larger start-up amounts experienced by the controller, which in turn, affects the outputs of the system. The boundary layer is perfectly satisfied as seen in the left-hand side of Figure 5.27.

### 5.5 Discussion

As seen through the several examples of SISO linear and nonlinear systems, and MIMO nonlinear systems, the modified control input formulation is capable of handling the presence of an actuator time-delay, provided the time-constant of the actuator is given, which is usually the case, or well estimated. The biggest issue faced in the simulations of the update control law was
satisfying the boundary layer condition, which is crucial when utilizing SMC to ensure close-loop asymptotic stability. At a time-delay of 0.01 seconds, perfect tracking was observed and the boundary layer condition was satisfied. Note that this is true in the simulations shown in the work here, where the sampling time of the simulation is 100x the actuator time delay. Changing the sampling time adversely affects the results of the simulation. This will have to be further investigated and examined. At higher time-delays, of 0.1 seconds or greater, where the effect of the actuator is more significant, the results were inconclusive. Although perfect tracking and a satisfactory boundary layer were obtained in the example of the 3rd-order SISO nonlinear system, the latter was not observed in the following examples. This suggests the need to further investigate the proposed modification to the control input equation, and possibly develop an updated boundary layer condition in order to handle the proposed method.
6 Conclusion

The model-free sliding mode controller derived by Crassidis and Reis in [16] for SISO systems was successfully extended and modified for applications of MIMO systems. The controller was derived for both fully-actuated and underactuated MIMO systems, and able to achieve perfect tracking on all or the desired outputs, respectively. The control system is based solely on the order of the system, system state measurements, previous control inputs, and the bounds and shape of the control input gain matrix. In this manner, the controller is still considered model-free, since the function describing the behavior of the system can be altered, and perfect control can still be achieved without a need to modify the control law. Additionally, parameter uncertainty was well-handled by the controller. As it compares to other SMC methods, such as those shown in the literature review section, the model-free SMC scheme achieved comparable performance, while not requiring explicit knowledge of the mathematical model of the system.

The first example in Section 3.3, implemented the classical SMC method on a first-order MIMO system in order to gauge the derivation process of the SMC as compared to the model-free method. The control law in that example is model dependent, and needs to be modified if any changes to the mathematical model occur. By implementing a boundary layer, control effort chattering was eliminated, and closed-loop asymptotic stability was achieved, while still maintaining perfect tracking of the desired trajectory.

The next case examined the application of the model-free control law on square MIMO systems, on both first and second order systems. The derivation and implementation of the control law on fully-actuated MIMO systems is similar to that performed in [16]. Model parameters and the control input gain matrix can be uncertain, but it is assumed that the bounds
of the control input gains are known, which is a reasonable assumption. Additionally, the bounds need not be accurate. The control law can handle wide margins of gain bounds, since the switching gain is capable of handling uncertainties in all parameters. The cost of robustness is an increase in the control effort.

The final case involved extending the model-free SMC law to underactuated MIMO systems. Since there are fewer inputs than outputs, the control input gain matrix is not square and therefore not invertible, which is a requirement for the derivation of the control law. In order to handle this issue, a transformation matrix was introduced, to essentially square the control input gain matrix, and allow for the derivation of the control law. Since perfect tracking cannot be achieved on all states simultaneously, the transformation matrix allowed for the choice of tracking certain outputs more than others. This method was then applied on several systems, including a single-input nonlinear 2 mass-spring-damper system, and a quadrotor. The former achieved perfect tracking on the desired output in all states, including the state with no direct control input, although control effort was maximized. Additionally, the latter also observed nearly perfect position tracking throughout, however, certain outputs and control effort both experienced high frequency activity. The reason behind this is thought to be the aggressiveness of the controller to ensure outputs perfectly track the entire trajectory, as oppose to merely settling at the desired final value.

Section 5 concluded the work by examining the effects of an actuator-induced time-delay on the model-free control system. As seen in the Section 5.1, simulating the presence of actuator delays had an adverse effect on the classical SMC technique, especially when the time delay exceeded 0.1 seconds. This was also true in the model-free application of SMC. Therefore, modifications had to be implemented to the derivation process to account for the presence of the
time delay. The modified control law was capable of handling the presence of time-delays, although the results were inconsistent at 0.1 seconds worth of time delay, in some cases. The cases examined included a SISO system, and both a fully-actuated and underactuated MIMO system, all of which observed perfect tracking.

6.1 Future Work

There are several ways the model-free SMC system can be improved. One thing that was present throughout all of the simulations performed was an algebraic loop. The reason this occurs is due to the need for the highest-order state to be fed into the formulation of the control law. In reality, this is not an issue since state measurements will be present from startup and available for the controller. However, it still limits further testing and development when it occurs in simulation, and is worth examining.

Another way the controller can be made more robust and adaptive, is to get better system parameter and control input gain estimates during operation, with the use of methods like online parameter estimation, which would significantly reduce control effort. Additionally control parameters like $\lambda$ and $\eta$ can be made time-varying to improve the control system’s performance and achieve asymptotic stability regardless of any system changes during operation.

Additionally, effects of larger actuator time delays can be investigated. Since the exact value of the time delay may not always be readily available, or even accurate, the control law can be modified to handle uncertainties in the actuator time delays as well. It can also be worth considering sensor delays as well, since measurements of all of the system’s states are necessary for the derivation of the control law, time lag in the input from sensors may cause some issues in performance.
Finally, the problem of high frequency activity of the controller and certain states, observed in the application of the control system on the quadrotor model, can be resolved by dialing back the controller gains and further optimizing the transformation matrix, in order to reduce the aggressive behavior of the control system.

6.2 Applications

The proposed model-free SMC method for MIMO applications was shown to be applicable to a wide range of systems, as long as the order of the system is known, state measurements are available for the controller, the shape of the control input gain matrix is known, and estimates can be made of the gain bounds. In cases where an actuator time-delay might be thought to have adverse effects on performance, the control law can be updated accordingly. With no requirement for the mathematical model of the system under control, this model-free SMC scheme can be very powerful, especially in controlling complex systems with no accurate model. Additionally, systems with models that might slightly change over time, or contain uncertain parameters, can be robustly controlled using the model-free sliding mode controller, since it does not rely on an explicit mathematical model.
7 References


